NATURA CUPIDITATEM INGENUIT HOMINI VERI VIDENDI
Marcus Tullius Cicero
(Природа наделила человека стремлением к познанию истины)

Мысли
Об
Истине

Альманах «МОИ»
Электронное издание, **ISBN 9984-688-57-7**

Альманах «Мысли об Истине» издается для борьбы с лженаукой во всех ее проявлениях и в поддержку идей, положенных в основу деятельности Комиссии РАН по борьбе с лженаукой и фальсификацией научных исследований. В альманахе публикуются различные материалы, способствующие установлению научной истины и отвержению псевдонаучных заблуждений в человеческом обществе.

Альманах издается с 8 августа 2013 года
Настоящая версия тома выпущена **2016-06-25**

© 2014 Марина Ипатьева (оформление и комментарии)

Предисловие к пяти выпускам альманаха МОИ

Начиная с этого номера нашего Альманаха, я опять создаю блок выпусков (№14, №15, №16, №17, №18), образующих одно целое, а именно – публикацию двух знаменитых книг Роджера Пенроуза с комментариями Валдиса Эгле. Эти книги для нас очень важны, потому что они теоретически доказывают, что невозможно сделать то, что Веданской теорией сделано.

Четыре года назад эти книги уже были опубликованы Валдисом Эгле в Интернете и остаются доступными также и сейчас. Так что я предпринимаю только их перепечатку. Необходимость переиздания я почувствовала по нескольким причинам.

Во-первых, в издании 2010 года материал был распределен по девяти файлам: PENRO1, PENRO2, PENRO3, PENRO4, PENRO5 (книга «Новый разум короля») и PENRS1, PENRS2, PENRS3, PENRS4 (книга «Тени разума»). В то время существовали ограничения на величину файлов на тех серверах, которыми В.Э. пользовался для публикации, поэтому материал приходилось разбивать на небольшие порции. Теперь этих ограничений уже нет, и я могу разбить материал вместо девяти на всего пять файлов. (Внутри, однако, сохраняю и метки первоначального деления на девять организационных единиц).

Во-вторых, издание 2010 года было ориентировано на чтение при помощи *MS Word* у себя в компьютере после того, как файлы (DOC) скачаны с Интернета (только в таком режиме отрабатывали все многочисленные перекрестные ссылки в этих книгах). В настоящем издании эти ссылки переориентированы на отработку в Интернете – без необходимости предварительного скачивания всех тех файлов, на которые данный файл ссылается. (Правда, при этом частично теряется точность ссылок.[1] Но у читателя есть возможность пользоваться как настоящим изданием в Интернете, так и изданием 2010 года с предварительным скачиванием).

В-третьих, ввиду чрезвычайной важности для Веданской теории книг Пенроуза (и комментариев В. Эгле к ним), мне хотелось включить этот материал в орбиту альманаха МОИ. Не будет ничего плохого в том, что эти книги будут существовать в Интернете в двух разновидностях.

При перепечатке этих книг я почти не присоединяла новых (своих) комментариев к ним. Комментарии В.Э., написанные в 2010 году и в начале 2011, в общем-то вполне достаточны (для того, что можно вложить в сами книги Пенроуза).

Однако более широкое обсуждение затронутых Пенроузом вопросов будет предпринято в разных выпусках Альманаха, начиная уже с №13).

Пенроуз неумело организовал ссылки в своих книгах. Ему нужно было разбить материал на небольшие параграфы (не больше страницы), перенумеровать их и ссылаться на эти номера. Такие ссылки остались бы в силе при любом издании и любом распределении материала по страницам. Вместо этого он ссылается на страницы (!!!) своего первого издания. Русские переводчики заменили страницы английского издания на страницы русского издания (всегда ли правильно – это еще вопрос). Валдис Эгле в своем издании сохранил номера страниц русского издания, но проставил под ними ссылки гипертекста на свои файлы. У меня уже нет возможности делать такие ссылки гипертекста, поэтому я могу либо перестроить всю систему ссылок на страницы выпусков МОИ, либо оставить (бессмысленную) ссылку на страницу русского бумажного текста. Первое чрезвычайно трудоёмко и неизвестно, будет ли вообще кто-нибудь ходить по этим ссылкам. Поэтому я такую работу выполнила лишь частично. Таким образом, если за номером страницы указано, что это МОИ, то это страница моего альманаха, а если не указано, то это страница бумажного издания. А всё было бы чрезвычайно просто, если бы Пенроузу хватило ума ссылаться на внутренние единицы текста, как это повсюду делаю я.

Марина Ипатьева

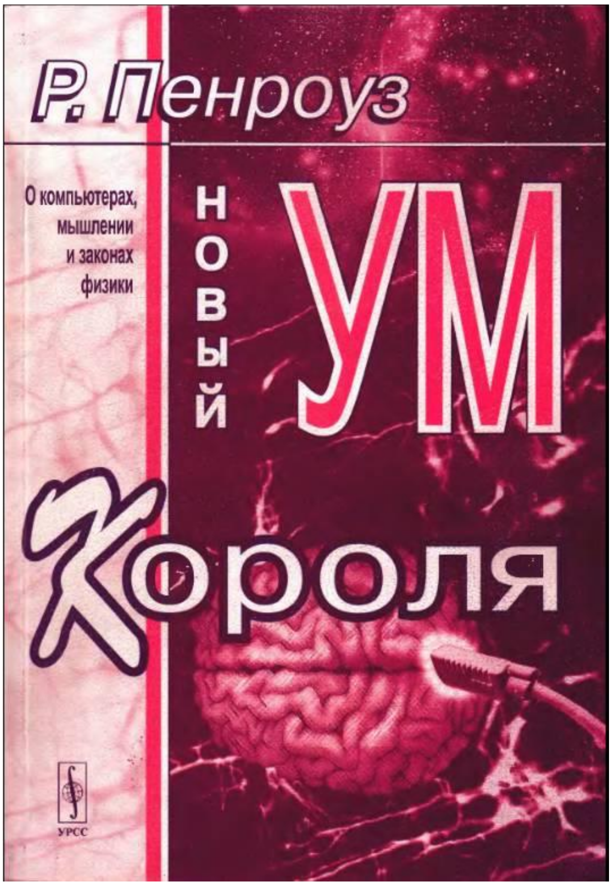
28 марта 2014 года

---

[1] Не отрабатывают ссылки из одного PDF файла на закладку (bookmark) другого PDF файла. При нынешнем уровне технологий это возможно только между HTML файлами. Но HTML файлы не переносимы – то есть, они не выглядят одинаково, откуда бы на них не смотрели. А я хочу, чтобы номер альманаха МОИ был всегда одинаков – такой, каким я его сделала – поэтому пользуюсь только PDF файлами. Жаль, конечно, что ради этого приходится терять точность ссылок, но переносимость файла для меня важнее. И теряется ведь только точность автоматического перехода по ссылке – то есть то, чего в бумажных изданиях нет и подавно. А то, что было возможно в бумажных изданиях, у нас сохраняется. Так что будем довольствоваться этим.

Файл PENRO1

http://vekordija.narod.ru/R-PENRO1.PDF



Передняя обложка книги

Предисловие в Векордии

Хотя эта книга написана Пенроузом раньше, чем «Тени разума» (только «Вступление» позже), в Векордию я ее включаю, когда Первый том «Теней» {PENRS1 = МОИ № 17} у меня уже готов и стоит в Интернете. Это отражается на характере комментариев. Первоначальные комментарии там, а здесь продолжение. Правда, большого значения это не имеет, но всё же – пусть читатель знает.

Валдис Эгле

26 августа 2010 года

Примечание МОИ о ссылках (2016-06-25)

Общеизвестно, что система ссылок должна быть организована по внутренним логическим единицам текста, а не по внешним номерам страниц. На евангелие ссылаешься, например, «Луки, 14:33», а на Энеиду «IV.625» – и эти ссылки действительны при всех изданиях на всех языках. Пенроуз же избрал систему ссылок (на страницы первого английского издания), которую иначе, как «идиотской» не назовешь. Из-за нее при переносе книг Пенроуза сначала в Векордию, потом в альманах МОИ было потрачено громадное количество усилий, но ссылки всё равно остаются запутанными и не до конца решенными. Но больше тратить времени на глупость Пенроуза я не могу, и оставляю ссылки в том виде, до какого мне их удалось довести. Извинения читателям.

Роджер Пенроуз. «Новый Разум Короля»

Посвящаю эту книгу светлой памяти моей дорогой матери, почившей прежде, чем эта книга увидела свет

Обращение к читателю

Как читать математические формулы

В некоторых частях этой книги я решился прибегнуть к математическим формулам. Меня не устрашило известное предостережение, что каждая формула в книге сокращает вдвое круг читателей. Если вы, Читатель, испытываете ужас перед формулами (как большинство людей), то я вам могу порекомендовать способ, который и сам часто использую, когда приличия нарушаются таким грубым образом. Способ заключается, более или менее, в том, чтобы полностью проигнорировать строку с формулой, сразу переводя взгляд на следующий за ней текст! На самом деле, конечно же, не совсем так: надо одарить формулу пытливым, но не проникающим взглядом, а затем двинуться вперед. Некоторое время спустя, почувствовав бóльшую уверенность в своих силах, можно вернуться к отвергнутой формуле и попытаться ухватить основные идеи. Текст, сопровождающий формулу, поможет вам понять, что в ней важно, а что можно спокойно проигнорировать. Если же этого все-таки не случилось, то смело оставляйте формулу и больше о ней не вспоминайте.

Благодарности

Многие помогали мне, тем или иным способом, в написании этой книги. Всем им я очень признателен. Для начала упомяну сторонников теории сильного ИИ (в особенности тех, которые выступали в телевизионной программе BBC), чьи радикальные идеи об искусственном интеллекте привлекли много лет назад мое внимание к этой теме. (Однако если бы я мог предвидеть заранее тот объем работы, который будет сопряжен с написанием этой книги, я вряд ли бы, думаю, начал.)

Многие скрупулезно читали отдельные части рукописи и высказывали мне свои идеи по ее улучшению. Им я приношу свою признательность. Это Тоби Бэйли, Давид Дойч (который мне очень помог в проверке описания машин Тьюринга), Стюарт Хампшир, Джим Хартли, Лэйн Хагстон, Ангус МакИнтир, Мэри Джэйн Моват, Тристан Неедман, Тед Ньюман, Эрик Пенроуз, Тоби Пенроуз, Вольфганг Риндлер, Энгельберт Шукинг и Дэннис Шьяма. Я очень благодарен Кристоферу Пенроузу за детальную информацию о множестве Мандельброта, а также Джонатану Пенроузу за сведения о шахматных компьютерах. Выражаю мою особую благодарность Колину Блэйкмору, Эрику Харту и Дэвиду Хьюбелу, которые внимательно прочитали главу 9, в предмете которой я, очевидно, совсем не специалист. Однако они – как и все остальные, которых я благодарю, – не отвечают за ошибки, если таковые сохранились. Я благодарен NSF[2] за поддержку по контракту DMS 84-05644, DMS 86-06488 (университет Райса, г. Хьюстон, где проходили многие лекции, частично легшие в основу этой книги), PHY 86-12424 (университет г. Сиракузы, где я участвовал во многих ценных обсуждениях по квантовой механике). Я премного обязан Мартину Гарднеру за его великодушное предложение написать предисловие к моей книге, а также за его ценные комментарии. Особенно благодарю мою дорогую Ванессу за ее вдумчивую и детальную критику некоторых глав, за неоценимую помощь с библиографией, а также, что

[2] NSF – аббревиатура с англ. *National Science Foundation* (Национальный Научный Фонд). – *Прим. ред.*

совсем немаловажно, за ее терпение, когда я был совсем невыносим – и за ее глубокую любовь и поддержку, когда я в этом особенно нуждался.

Предисловие

Для многих великих физиков и математиков написать книгу, понятную не только профессионалам – дело трудное, если не сказать невозможное. И вплоть до сего времени иным могло бы показаться, что Роджер Пенроуз, один из наиболее компетентных и плодотворно работающих физиков-теоретиков во всем мире, относится как раз к такой категории ученых. Но даже для тех из нас, кто был знаком с его популяризаторскими статьями и лекциями и не разделял подобного мнения, появление превосходной книги для широкого круга читателей, ради которой он оторвал от работы часть своего времени, стала приятным сюрпризом. И я не сомневаюсь, что этой книге в будущем уготовано стать классической монографией.

Хотя в различных главах своей книги Пенроуз затрагивает и теорию относительности, и квантовую механику, и космологию – главным объектом его рассуждений является так называемая психофическая[3] проблема «ум–тело». Десятилетиями сторонники теории «сильного ИИ» (искусственного интеллекта) пытались убедить нас, что не пройдет и одного–двух веков (а некоторые опускали эту планку даже до пятидесяти лет!), как электронные компьютеры полностью сравняются по своим возможностям с человеческим мозгом. Находясь под впечатлением прочитанных в юности научно-фантастических книг и будучи убежденными в том, что наши мозги – это просто «компьютеры, сделанные из мяса» (как выразился однажды Марвин Мински), они считали несомненным, что удовольствие и боль, восприятие прекрасного и чувство юмора, сознание и свобода воли – все эти способности возникнут у электронных роботов сами собой, как только управляющие ими алгоритмы обретут достаточную степень сложности.

Но некоторые методологи науки (в особенности Джон Серл, чей мысленный эксперимент со знаменитой китайской комнатой Пенроуз очень подробно разбирает в одной из глав) с этим решительно не согласны. В их представлении компьютер по существу ничем не отличается от обычных механических калькуляторов, в которых арифметические действия выполняются посредством колесиков, рычажков или иных приспособлений, позволяющих передавать сигналы. (За основу компьютера с таким же успехом можно взять, например, маленькие перекатывающиеся шарики или текущую по системе труб воду.) Поскольку электричество движется по проводам быстрее, чем любая иная форма энергии (за исключением света), электрические устройства могут оперировать символами с большей скоростью, что позволяет им выполнять чрезвычайно громоздкие и сложные задачи. Но «осознает» ли компьютер свои действия в большей мере, чем это доступно обычным деревянным счетам? Сегодня компьютеры могут играть в шахматы на уровне гроссмейстеров. Но «понимают» ли они эту игру лучше, чем машина для «крестиков-ноликов», собранная группой компьютерных хакеров из поломанных игрушек?

Книга Пенроуза является самой мощной атакой на теорию сильного ИИ из всего написанного до сих пор.[4] За несколько прошедших столетий было высказано немало возражений против понимания мозга как машины, управляемой общеизвестными законами физики; но доводы Пенроуза более убедительны, ибо они базируются на недоступной для его предшественников информации. Эта книга открывает нам другого Пенроуза – не только математика и физика, но и философа высокого уровня, не отступающего перед проблемами, которые современные философы слишком легко сбрасывают со счетов как бессмысленные.

К тому же Пенроуз, вопреки всё более настойчивым возражениям небольшой группы физиков, имеет смелость отстаивать позиции здорового реализма. В его представлении реальна не только вселенная, но и математическая истина, непостижимым образом ведущая свое собственное независимое и вечное существование. Подобно Ньютону и Эйнштейну, Пенроуз испытывает благоговейный трепет и чувство смирения как перед физическим миром, так и перед Платоновым царством чистой математики. Выдающийся ученый в области теории чисел Пол Эрдос любит говорить «о божественной книге», в которой записаны все лучшие доказательства.[5] И математикам иной раз приоткрывается та или иная ее страница. Моменты прозрения, когда математик или физик внезапно вскрикивает «Ага!», по мнению Пенроуза, не могут явиться

[3] **В.Э.:** психофизическая?

[4] **В.Э.:** Подчеркнуто мной.

[5] **В.Э.:** Не доказательства, а истины.

«результатом сколь угодно сложных вычислений»: в эти мгновения разум соприкасается с объективной истиной. Возможно ли, вопрошает Пенроуз, что мир «идей» Платона и реальный физический мир (который физики сегодня всё больше «растворяют» в математике) – на самом деле тождественны?

Большое внимание в книге Пенроуза уделяется знаменитой фрактальной структуре, называемой множеством Мандельброта в честь ее первооткрывателя Бенуа Мандельброта. Хотя в статистическом смысле такие объекты обладают свойством самоподобия, которое выявляется при увеличении отдельных частей, их бесконечно причудливые очертания постоянно меняются самым непредсказуемым образом. Пенроузу кажется непонятным, как можно сомневаться в том, что эти экзотические структуры существуют не менее «реально», чем гора Эверест, и могут быть исследованы точно так же, как исследуются джунгли.

Пенроуз принадлежит к постоянно пополняющейся группе ученых, которые считают, что Эйнштейн не был упрямым или, тем более, бестолковым, когда однажды, ссылаясь на свой «левый мизинец», он провозгласил неполноту квантовой механики. Чтобы подтвердить справедливость этого утверждения, Пенроуз увлекает читателя в головокружительное путешествие, в ходе которого мы знакомимся с комплексными числами, машинами Тьюринга, теорией сложности, поразительными парадоксами квантовой механики, формальными системами, (теоремой) неразрешимости Гёделя, фазовыми и гильбертовыми пространствами, черными и белыми дырами, излучением Хокинга, энтропией, строением мозга – и множеством других вопросов, занимающих сегодня умы ученых. «Осознают» ли кошки и собаки свое «я»? Могут ли в теории существовать передатчики материи, способные переместить человека из одного места в другое на манер астронавтов из сериала *Звездный Путь*? Насколько полезно нам – с точки зрения выживания – возникшее в ходе эволюции сознание? Существует ли структура более общая, чем квантовая механика, где бы нашлось естественное объяснение направлению времени и различиям между правым и левым? Важны ли законы квантовой механики, а может и некие более «тонкие» законы, для деятельности разума?

На два последних вопроса Пенроуз дает положительный ответ. Его знаменитая теория «твисторов» – абстрактных геометрических объектов, действующих в многомерном комплексном пространстве, которое лежит в основе обычного пространства-времени – носит чересчур узкоспециализированный характер, чтобы быть включенной в эту книгу. Она стала результатом его двадцатилетних усилий проникнуть в область более глубокую, чем квантовые поля и частицы. Прибегая к своей четырехступенчатой классификации теорий – превосходных, полезных, пробных и тупиковых, – Пенроуз скромно поместил теорию твисторов в разряд пробных, вместе с суперструнами и другими теориями великого объединения, которые сейчас вызывают острые дискуссии в научной среде.

С 1973 года Пенроуз возглавляет кафедру Рауза Болла в Оксфордском университете. Это тем более заслуженно, что В.У. Рауз Болл был не только выдающимся математиком, но еще и фокусником-любителем, настолько увлеченным занимательной математикой, что однажды он даже написал на эту тему ставшую классической книгу *Математические эссе и развлечения*[6]. Пенроуз разделяет эту страсть Болла к играм. В юности он придумал «невозможный объект», состоящий из трех стержней. (Невозможный объект – это изображение цельной фигуры, которая не может существовать из-за наличия в ней внутренне противоречивых элементов.) Вместе со своим отцом Лайонелом, генетиком по профессии, он превратил свой невозможный объект в *Лестницу Пенроуза*, структуру, использованную Морицем Эшером на двух известных литографиях: *Идущие вверх и идущие вниз* и *Водопад*. В один прекрасный день, когда Пенроуз лежал в кровати, с ним случился, как он сам называет это, «приступ сумасшествия», когда ему явственно представился невозможный объект в четырехмерном пространстве. Если бы существо из четырехмерного мира наткнулось на эту штуку, шутит Пенроуз, оно наверняка воскликнуло бы: «Боже мой, что это такое!?»

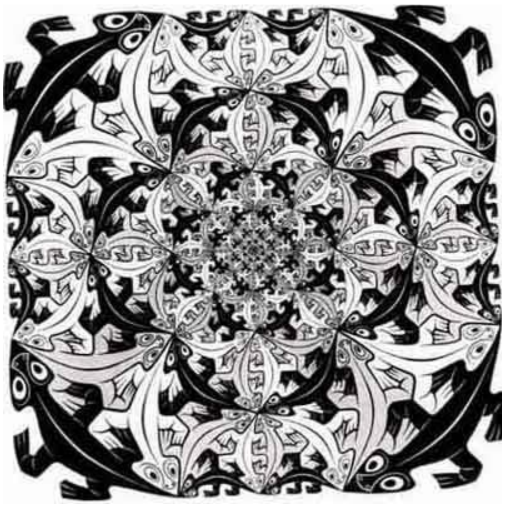
Работая в 1960-х годах вместе со своим другом Стивеном Хокингом над проблемами космологии, он сделал свое самое, наверное, известное открытие. Если теория относительности выполняется «до самого конца», то в каждой черной дыре должна существовать сингулярность, где законы физики теряют свою силу. Но даже это достижение отошло в последние годы на второй план после того как Пенроуз предложил конструкцию из «плиток» двух видов, которыми

---

[6] Rouse Ball, W.W. and Coxeter, H.S.M., *Mathematical Recreations and Essays*. Macmillan, London, 1959. (Рус. пер.: Болл У., Коксетер Г. Математические эссе и развлечения. М.: Мир, 1986). – *Прим. ред*.

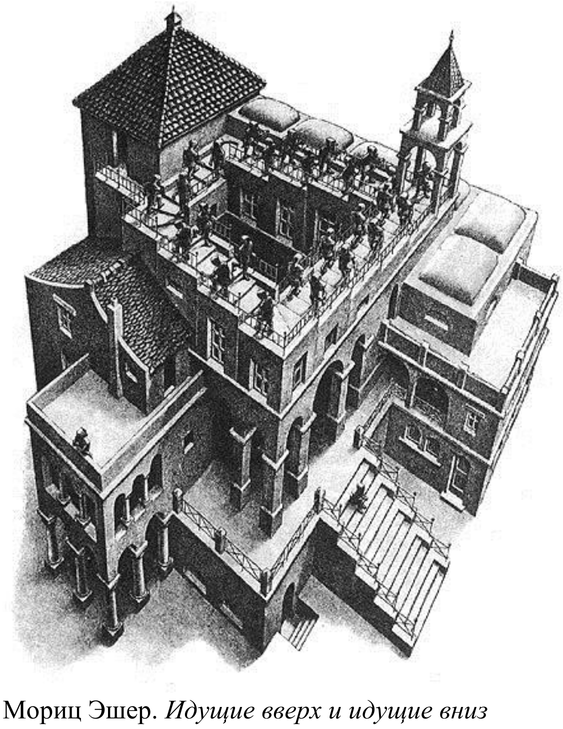
можно покрыть всю плоскость подобно мозаике Эшера – только непериодическим образом. (Об этих удивительных фигурах вы можете узнать подробнее в моей книге *От мозаик Пенроуза к надежным шрифтам*[7].) Пенроуз изобрел, или, скорее, открыл их, даже не предполагая, что когда-нибудь они могут кому-то пригодиться. К всеобщему изумлению оказалось, что трехмерные аналоги этих фигур могут служить основой для новой необычной формы материи – «квазикристаллов». Сейчас изучение «квазикристаллов» превратилось в одну из наиболее активных областей исследований в кристаллографии. Это, безусловно, самый впечатляющий пример того, как в наши дни математические игры могут иметь совершенно неожиданные практические приложения.

Достижения Пенроуза в математике и физике – а я упомянул только незначительную их часть – рождаются из постоянно присутствующего в его душе ощущения тайны и красоты бытия. Мизинец «подсказывает» ему, что человеческий мозг представляет собой устройство более сложное, чем набор крошечных проводков и переключателей. Фигура Адама в прологе и эпилоге этой книги в определенном смысле служит символом зарождения разума в ходе неторопливого развития осознающей себя жизни. В нем я тоже вижу Пенроуза – мальчика, сидящего в третьем ряду, позади признанных корифеев в области ИИ, – который не боится высказать им вслух свое мнение, что их «короли-то голые»[8]). Юмор присущ многим высказываниям Пенроуза, но это утверждение – отнюдь не шутка.



Одна из мозаик Эшера

Мартин Гарднер



Мориц Эшер. *Идущие вверх и идущие вниз*

Мориц Эшер. *Водопад*

[7] Gardner, M. *Penrose tiles to trapdoor ciphers*. W.H. Freeman and Company, New York, 1989. (Рус. пер.: Гарднер М. *От мозаик Пенроуза к надежным шрифтам*. М.: Мир, 1993). – *Прим. ред*.

[8] В оригинале название книги *The Emperor's New Mind* перекликается с названием известной сказки Г.-Х. Андерсена *The Emperor's New Clothes* – Новый наряд короля. – *Прим. ред*.

Вступление

Книга *Новый ум короля*, впервые изданная в 1989 году, стала моей первой серьезной попыткой написать научно-популярное произведение. Приступая к созданию этой книги, я, помимо всего прочего, ставил целью рассказать в максимально доступной форме о значительном прогрессе физической науки, достигнутом в познании законов окружающего нас мира. Но это не просто обзор научных достижений. Я еще и пытаюсь указать на целый ряд принципиальных трудностей, которые стоят перед наукой на ее пути к конечной цели. В частности, я утверждаю, что явление сознания не может быть описано в рамках современной физической теории.

Это явно противоречит довольно устоявшемуся пониманию сущности научного подхода, согласно которому все аспекты умственной деятельности (включая, в том числе, и сознание) – не более, чем результат вычислений, происходящих в мозге; соответственно, электронные компьютеры должны быть потенциально способны к сознательному восприятию, которое возникло бы само собой[9] при наличии достаточной мощности и соответствующих программ. Я постарался по возможности беспристрастно аргументировать свое несогласие с таким взглядом, указывая на то, что проявления сознательной деятельности мозга не могут быть объяснены в вычислительных терминах и – более того – с позиций современного научного мировоззрения в целом.[10] Однако я ни в коем случае не утверждаю, что понимание этого феномена невозможно в рамках научного подхода – просто современная наука еще не достигла уровня, необходимого для решения такой задачи.

Когда я писал эту книгу, мне трудно было вообразить, сколь бурной окажется реакция на изложенные в ней мысли – причем не только из лагеря убежденных сторонников «компьютерной» модели разума, но и со стороны тех, кто считает научный метод недопустимым для изучения сознания. Я нисколько не сомневаюсь, что попытка затронуть чью-то личную философскую концепцию сознания – как и религиозные воззрения – может оказаться делом довольно рискованным. Но насколько щекотливой бывает подчас эта тема – я едва ли мог представить себе в полной мере.

Мои рассуждения в том виде, в котором они представлены в книге, направлены на достижение двух целей. Первая из них – это стремление показать, опираясь главным образом на результаты, полученные Гёделем (и Тьюрингом), что математическое мышление – а, следовательно, и умственная деятельность в целом – не может быть полностью описано при помощи чисто «компьютерной» модели разума. Именно эта часть моих умозаключений вызывает у критиков наиболее настойчивые возражения. Вторая цель – показать, что сегодня в физической картине мира есть существенное «белое пятно», а именно: отсутствует «мостик» между субмикроскопическим уровнем квантовой механики и макромиром классической физики. С моей точки зрения, теория, которая однажды восполнит этот пробел, должна будет в значительной степени помочь понять физические основы феномена сознания. Более того, в этой искомой

[9] **В.Э.:** Здесь у Пенроуза противоречие: так это сознание у компьютеров должно возникнуть «само собой» – или «при наличии соответствующих программ»? (Это ведь вещи прямо противоположные!) «Само собой» оно никогда не возникнет. «При наличии соответствующих программ» оно не «возникнет», а будет сделано, осуществлено. Вообще уже само построение предложения свидетельствует о том, что у Пенроуза нет в голове представления о тех взглядах, которые являются действительной альтернативой его воззрениям. Он всё время борется с какими-то наивными представлениями (не знаю уж: действительно существующими у кого-то или самим Пенроузом выдуманными из-за неправильного понимания?). Согласно этим наивным представлениям (с которыми Пенроуз борется) «сознание» должно «возникнуть» спонтанно – вследствие одного лишь достижения определенной сложности устройства. Но ведь такое мнение не настоящий противник для Пенроуза! А против настоящего противника (например, взглядов, представляемых мною) он не выдвигает никаких аргументов – просто вообще его не касается – видимо, позицию этого настоящего противника он вообще не знает.

[10] **В.Э.:** Вообще это само по себе чрезвычайно рискованное, опрометчивое и легко уязвимое заявление. Как нам отличить две ситуации: 1) «проявления сознательной деятельности мозга не могут быть объяснены с позиций современного научного мировоззрения» и 2) «проявления сознательной деятельности мозга Пенроуз не умеет объяснить с позиций современного научного мировоззрения»? Какие критерии в принципе могут быть выдвинуты, чтобы убедить, что имеет место не (2), а (1)? Доказать несуществование чего-то всегда намного сложнее, чем существование. А чтобы доказать, что имеет место (2), достаточно объяснить «проявления сознательной деятельности с позиций современного научного мировоззрения». (Что Веданская теория и делает).

области физики должно быть заложено нечто выходящее за рамки только вычислительных действий.

За десятилетие, прошедшее с момента первого издания книги, наука добилась целого ряда ошеломляющих успехов. Про некоторые из них я бы хотел вкратце рассказать здесь с тем, чтобы у читателя сложилось определенное представление о моем вӥдении современного состояния этих исследований. Сперва рассмотрим, насколько важна теорема Гёделя для критики выдвинутых мной положений. Если попытаться изложить в двух словах суть этой теоремы (справедливость которой не оспаривается)[11], то она будет выглядеть следующим образом. Пусть мы располагаем какой-нибудь вычислительной процедурой *P*, позволяющей нам формулировать математические утверждения (для определенности договоримся, что это будут утверждения какого-то одного вида, аналогичные, допустим, знаменитой теореме Ферма (см. §2.7)). Тогда, если мы готовы считать правила процедуры *P* надежными – в том смысле, что мы будем полагать всякое математическое утверждение, полученное при помощи этой процедуры, неоспоримо верным, – то равным образом мы должны принимать и неоспоримую справедливость некоторого утверждения *G*(*P*), которое лежит за пределами действия правил процедуры *P* (см. §4.3). Таким образом, как только мы научились автоматизировать некоторую часть нашего математического мышления, у нас сразу же появляется понимание, как выйти за его границы.[12] В моем представлении это однозначно свидетельствует о том, что математическое понимание содержит определенные элементы, которые не могут быть полностью сведены к вычислительным методам. Но многие критики остались при своих убеждениях, указывая на различные возможные «тонкие места» в этих логических построениях. В моей следующей книге *Тени разума*[13] я постарался ответить на все подобные возражения и привел ряд новых аргументов в пользу своей точки зрения. Тем не менее споры всё еще продолжаются.[14]

Одна из причин, мешающих людям признать прямое отношение, которое имеет теорема Гёделя к нашему математическому мышлению, заключается в том, что в рамках обычной ее формулировки утверждение *G*(*P*) не представляет интереса с математической точки зрения. Мало того: оно еще и чрезвычайно сложно для понимания в качестве математического выражения. Соответственно, даже математики предпочитают не «связываться» с подобными выражениями. Однако, существует ряд примеров утверждений гёделевского типа, которые легко доступны пониманию даже для тех, чье знакомство с математической терминологией и системой записи ограничивается рамками обычной арифметики.

Особенно впечатляющий пример попался мне на глаза уже после того, как была опубликована эта книга (а также *Тени разума*). Это произошло на лекции Дэна Исааксона в 1996 году. Речь шла об известной теореме Гудстейна.[15] Данный пример кажется мне настолько поучительным, что я хотел бы рассмотреть его здесь целиком, дабы читатель имел возможность непосредственно познакомиться с теоремами гёделевского типа[16].[17]

---

[11] **В.Э.:** Кем не оспаривается? ☺

[12] **В.Э.:** Я давно пытаюсь уловить и понять до конца, – ну ПОЧЕМУ Пенроуз приходит к своим взглядам о «невычислимости» математического мышления в частности и сознания вообще, – но это мне никак не удается по-настоящему. Так и здесь: вообще-то, если у нас имеется процедура *P*, что-то делающая по интеллекту (в том числе «*позволяющая нам формулировать математические утверждения*»), то почему «выход за ее границы» свидетельствует «*о том, что математическое понимание содержит определенные элементы, которые не могут быть полностью сведены к вычислительным методам*»? Ну, была у меня программа *P*, потом нашел более сильную программу – ну и что? Пенроуз как-то «зациклился» на этой «теореме Гёделя» (к тому же неверно им интерпретированной).

[13] *Shadows of the Mind*, Oxford University Press, 1994; pb. Vintage, 1995.

[14] Заинтересованный читатель может ознакомиться с критическими замечаниями и моими ответами на них в *Behavioral and Brain Sciences,* 13 (4) (1990), 643–705 и в *Psyche* (MIT Press), 2 (1996), 1–129. Последний из этих материалов можно найти на веб-сайте http://psyche.cs.monash.edu.au/psyche-index-v2-l.html; я рекомендую прочитать приведенные мной возражения (озаглавленные *Beyond the Doubting of a Shadow*) на критические замечания по этому вопросу перед тем, как приступить к чтению книги *Тени разума*. Следующим источником дополнительной информации может служить моя работа *The Large, the Small and the Human Mind* (Cambridge University Press, 1997).

[15] Goodstein, R.L., *On the restricted ordinal theorem*. Journal of Symbolic Logic, 9 (1944), 33–41.

[16] См. также Penrose, R., *On understanding understanding*. International Studies in the Philosophy of Science, 11 (1997), 7–20.

[17] **В.Э.:** Вот, слава богу! На конкретных примерах разбираться гораздо лучше, чем общими словами!

Чтобы понять суть этой теоремы, рассмотрим любое целое положительное число, скажем, 581. Для начала мы представим его в виде суммы различных степеней числа 2:

$$581 = 2^9 + 2^6 + 2^2 + 1.$$

(Такая процедура применяется для формирования двоичного представления числа 581, а именно, приведения его к виду 1001000101, где единицы соответствуют тем степеням двойки, которые присутствуют в таком представлении, а нули – тем степеням, которых нет.) Далее можно заметить, что «показатели» в этом выражении – т.е. 9, 6 и 2 – могут быть, в свою очередь, представлены аналогичным образом ($9 = 2^3 + 1$, $6 = 2^2 + 2^1$, $2 = 2^1$); и тогда мы получим (вспоминая, что $2^1 = 2$)

$$581 = 2^{2^3+1} + 2^{2^2+2} + 2^2 + 1.$$

Здесь всё еще есть показатель больший, чем двойка – в данном случае это «3», – для которого тоже можно написать разложение $3 = 2^1 + 1$, так что в конце концов мы будем иметь

$$581 = 2^{2^{2+1}+1} + 2^{2^2+2} + 2^2 + 1. \qquad (1)$$ [18]

А теперь мы подвергнем это выражение последовательности чередующихся простых операций, которые будут

(а) увеличивать «основание» на единицу,

(б) вычитать единицу.

Под «основанием» здесь понимается просто число «2», фигурирующее в исходном выражении, но мы можем сделать то же самое и с бóльшими основаниями: 3, 4, 5, 6 … . Давайте посмотрим, что произойдет при применении операции (а) к последнему разложению числа 581, в результате которой двойки становятся тройками:

$$3^{3^{3+1}+1} + 3^{3^3+3} + 3^3 + 1,$$

(что дает – если выписать его в обычной форме – сороказначное число, начинающееся с 133027946...). После этого мы применяем (б) и получаем

$$3^{3^{3+1}+1} + 3^{3^3+3} + 3^3$$

(т. е. по-прежнему сороказначное число, начинающееся с 133027946 ...). Далее мы выполняем (а) еще раз и получаем

$$4^{4^{4+1}+1} + 4^{4^4+4} + 4^4 \qquad (2)$$

(это уже значительно большее число, состоящее из 618 знаков, которое начинается с 12926802...). Следующая операция – вычитание единицы – приводит к выражению

$$4^{4^{4+1}+1} + 4^{4^4+4} + 3 \times 4^3 + 3 \times 4^2 + 3 \times 4 + 3 \qquad (3)$$

(где тройки получаются по той же причине, что и девятки в обычной десятичной записи, когда мы получаем 9999, вычитая 1 из 10 000). После чего операция (а) дает нам

$$5^{5^{5+1}+1} + 5^{5^5+5} + 3 \times 5^3 + 3 \times 5^2 + 3 \times 5 + 3$$

(число, которое имеет 10923 знака и начинается с 1274...). Обратите внимание, что коэффициенты «3», которые возникают при этом, с необходимостью меньше, чем основание (в данном случае 5), и не изменяются с возрастанием последнего.[19] Применяя (б) вновь, имеем число

$$5^{5^{5+1}+1} + 5^{5^5+5} + 3 \times 5^3 + 3 \times 5^2 + 3 \times 5 + 2,$$

над которым мы опять производим последовательно действия (а), (б), (а), (б), ... и т.д., насколько возможно. Вполне естественно предположить, что этот процесс никогда не завершится, потому что каждый раз мы будем получать всё бóльшие и бóльшие числа. Однако это не так: как следует из поразительной теоремы Гудстейна, независимо от величины исходного числа (581 в нашем примере), мы <u>в конце концов получим нуль</u>![20]

Кажется невероятным, но это так. А чтобы в это поверить, я рекомендовал бы читателю самостоятельно проделать вышеописанную процедуру, для начала – с числом «3» (где мы раскладываем тройку как $2^1 +1$, что дает последовательность 4, 3, 4, 2, 1, 0); а затем – что более важно – попробовать то же самое с «4» (при этом стартовое разложение в виде $4 = 2^2$ приводит к

---

[18] **В.Э.:** Красные номера выражений здесь проставлены мною для дальнейшего разбора.

[19] **В.Э.:** В данном случае (когда показатель был равен базису – основанию) не возрастают не только коэффициенты, но и показатели степеней у базисов за ними.

[20] **В.Э.:** Нет! – в том виде, как задача сформулирована Пенроузом здесь, мы получим $-\infty$ (минус бесконечность). Чтобы получить ноль, надо оговорить, что вычисления проводятся только в положительных числах и останавливаются, когда надо из нуля вычесть единицу.

вполне закономерно возрастающему ряду 4, 27, 26, 42, 41, 61, 60, 84, ... , который доходит до числа из 121˙210˙695-ти знаков, после чего уменьшается вплоть до нуля!).

Но что кажется еще более удивительным: теорема Гудстейна фактически является теоремой Гёделя для той самой процедуры, которую мы изучали в школе под названием математической индукции, как было доказано в свое время Л. Кирби и Дж. Парисом.[21] Как вы, должно быть, помните, математическая индукция позволяет установить справедливость некоторого математического утверждения $S(n)$ для n = 1, 2, 3, 4, 5, ... Доказательство проводится в два этапа: сначала нужно проверить справедливость $S(1)$, а затем показать, что, если верно $S(n)$, то должно выполняться и $S(n+1)$. Приняв процедуру математической индукции за P, Кирби и Парис доказали, что тогда $G(P)$ может иметь смысл теоремы Гудстейна.

Следовательно, если мы считаем процедуру математической индукции достоверной (с чем едва ли можно не согласиться), то мы должны верить и в справедливость теоремы Гудстейна – несмотря на то, что при помощи одной лишь математической индукции доказать ее невозможно.

«Недоказуемость» теоремы Гудстейна, понимаемая в этом смысле, вряд ли может помешать нам убедиться в ее фактической справедливости. Наши интуитивные представления позволяют нам расширить действие тех ограниченных приемов «доказательства», которыми мы воспользовались ранее. В действительности сам Гудстейн доказал свою теорему, прибегнув к разновидности метода, который называется «трансфинитной индукцией». В контексте нашего изложения этот метод сводится к систематизации интуитивных ощущений, которые возникают в процессе знакомства с «причиной», по которой теорема Гудстейна и в самом деле верна. Эти ощущения могут родиться практически целиком за счет изучения некоторого числа частных случаев указанной теоремы. И тогда станет видно, как скромная незаметная операция (б) безжалостно «отщипывает» по кусочку от огромной башни «показателей» до тех пор, пока она не начинает постепенно таять и полностью исчезает, – хотя бы на это ушло и невообразимо большое число шагов.

Всё это говорит о том, что способность понимать никоим образом не может сводиться к некоторому набору правил. Более того, понимание является свойством, которое зависит от нашего сознания; и что бы не отвечало в нас за сознательное восприятие – это должно самым непосредственным образом участвовать в процессе «понимания». Тем самым, в формировании нашего сознания с необходимостью есть элементы, которые не могут быть получены из какого бы то ни было набора вычислительных инструкций; что, естественно, дает нам веские основания считать, что сознательное восприятие – процесс существенно «невычислимый».

* * *

2010.08.29 20:41 воскресенье

В.Э.: Здесь я прерву рассказ Пенроуза, чтобы объяснить, как всё это выглядит с точки зрения Веданской теории и почему для этой точки зрения слова Пенроуза выглядят перекошенными, неправильно интерпретирующими действительность и местами даже вообще неверными. (Как хорошо, что можно разбирать конкретный пример!).

Итак, дана задачка оперирования над числами и их представлениями в различных системах счисления. Это задачка «чисто программистская»: дан алгоритм (и по нему можно написать программу), получающий на входе число (типа *integer без знака*)[22] и состоящий из четырех шагов:

1. разложить исходное число (и далее – все показатели степеней) по степеням базиса 2;
2. получить новое число путем замены актуального базиса n на $n+1$ (операция (а));
3. вычесть из нового числа единицу (операция (б));
4. перейти к пункту (2).

---

[21] *Accessible independence results for Peano arithmetic*. Bulletin of the London Mathematical Society, 14 (1982), 285–93.

[22] Вообще в разных языках программирования обозначения типов данных немножко отличаются, хоть и существуют общие традиции. Бóльшую часть своей программистской жизни я программировал для ЕС ЭВМ (IBM/360/370) на Ассемблере (на нем был написан и Диспос); потом, для компьютеров IBM/PC – на «*Borland Pascal*». На Ассемблере (приблизительно) этот тип данных назывался *fixed* (число с фиксированной точкой); в борландовском «Паскале»: *word* или *byte*. Но эти названия плохо подходят для общего случая. Нам лучше всего было бы обозначать этот тип данных как *natural*, но я не помню, существует ли такое название в каком-нибудь реальном языке программирования.

Этот алгоритм будет работать заведомо бесконечно, так как в него не заложено никакое условие выхода из цикла шагов (2–4). По этому алгоритму может работать мозговая программа, и по нему может быть написана программа для промышленного компьютера.

Я любил называть свои программы различными именами, особенно заканчивающимися на «-ор» или «-ер». Продолжим эту традицию и назовем «гудстейнатором» программу, работающую по этому алгоритму. Итак, у нас могут быть гудстейнаторы для мозга (биологического компьютера) и гудстейнаторы для промышленных компьютеров.

Представим, что мне, как программисту, дана задача написать гудстейнатор для промышленного компьютера. Может показаться, что эта задача практически неосуществимая, так как очень скоро потребуются числа, гораздо большие, чем позволяет конструкция любого современного реального компьютера. Но опытный программист для данной задачи и не подумает пользоваться «встроенными» «числами», а создаст свои собственные представления чисел. Так я запросто могу в обычном компьютере держать числа с миллионами знаков, причем эти миллионы знаков – не само число, а базисы и показатели степеней, так что мой гудстейнатор сможет оперировать громаднейшими числами, хотя, конечно, в конце концов и его возможности представления чисел окажутся исчерпанными.

И, разумеется, еще скорее будут исчерпаны ресурсы времени: гудстейнатор будет работать долго – страшно долго; при многих входных *n* гораздо дольше, чем существует наша Вселенная с момента Большого Взрыва.

Но всё это не помешает нам изучить эту программу (и этот алгоритм) «абстрактно», т.е. на самом деле не выполняя ее (программу) и его (алгоритм). Это такая работа, которую делает каждый день всякий работающий программист; над каждой своей программой он сидит и думает: что будет происходить, когда я ее запущу?

Вот, и мы, давайте, выполним стандартную программистскую работу и подумаем: что будет происходить, если мы запустим гудстейнатор? На рис. VE1 показаны структуры данных гудстейнатора для пенроузовской формулы (1).

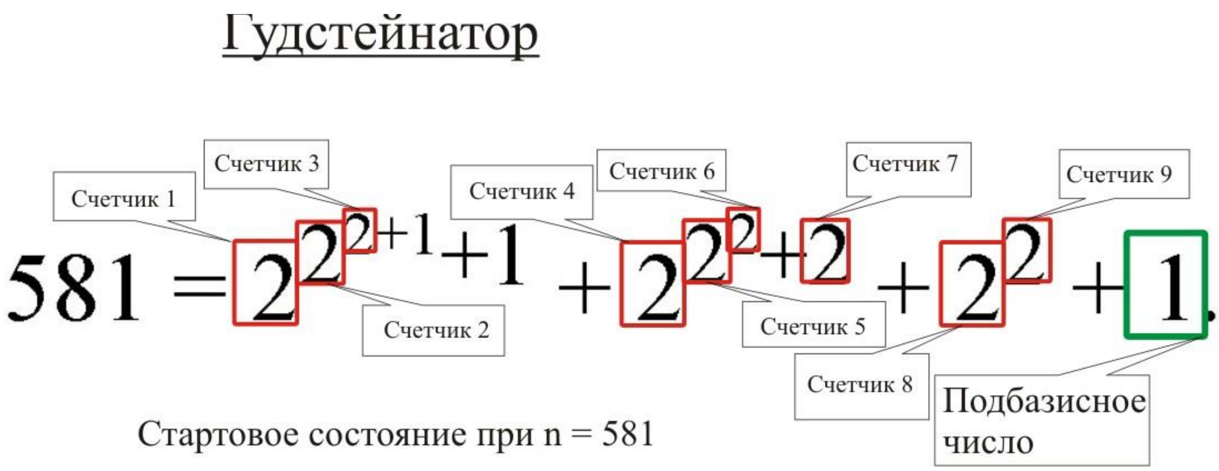


Рис. VE1. Структуры данных программы «Гудстейнатор»

В данном случае в структурах имеются девять переменных, названных «счетчиками» и одна переменная, названная «подбазисным числом». На каждом цикле работы гудстейнатора операция, обозначенная Пенроузом как (а), будет увеличивать все счетчики, а операция (б) вычитать единицу из подбазисного числа.

Ясно, что при другом входном *n* стартовое число счетчиков будет в общем случае другим. Счетчики образовывают каскады (так, в примере рисунка VE1 первый каскад составляют счетчики 1–3, второй каскад счетчики 4–7, а третий каскад счетчики 8–9). Каскады могут «ветвиться» (как на рис. VE1 второй каскад), образуя древовидные структуры. (Ясно, что структуры гудстейнатора должны строиться на списках;[23] это обычное дело – весь Диспос[24]

[23] Списком в программировании называется такая структура данных, в которой заранее не известно (даже максимальное) число элементов; элемент строится только тогда, когда он понадобился, и включается в структуру так, что ссылка на него (его адрес) отмечается в предыдущем элементе (или в корне списка,

работал только на списках, и вообще мои программы почти не используют другие виды структур).

Пока подбазисное число больше нуля, операция (б) не затрагивает каскады счетчиков, и счетчики спокойно растут. Очевидно, что коренной момент в работе гудстейнатора – это когда подбазисное число исчерпано, и нужно «занять» разряд у крайнего каскада счетчиков. Исследование этого процесса и есть центральное звено изучения работы гудстейнатора вообще. На пенроузовских формулах (2) и (3) виден один частный случай этого процесса (см. рис. VE2).

$$4^{4^{4+1}+1} + 4^{4^4+4} + \boxed{4^4}$$

$$4^{4^{4+1}+1} + 4^{4^4+4} + \boxed{3 \times 4^3 + 3 \times 4^2 + 3 \times 4 + 3}$$

Распад младшего каскада счетчиков при вычитании 1, когда подбазисного числа нет

Рис. VE2. Счетчик в показателе степени распадается на ряд счетчиков с подбазисными показателями степени

Изучив этот процесс, мы видим, что вычитание единицы при нулевом подбазисном числе разрушает крайний правый каскад счетчиков. Исчезает верхний счетчик-показатель, и вместо него появляется ряд счетчиков более низкого уровня, но при этом счетчиков уже нет в показателях степени. Детальное исследование всех возможных случаев здесь заняло бы слишком много места (оно и есть «доказательство теоремы Гудстейна»), но общий принцип ясен.

Вычитание единицы сначала уничтожает подбазисное число, потом, когда оно уничтожено, ударяет по крайнему правому каскаду, стаскивает счетчики с верхних «полок» показателей вниз, а потом и вовсе разлагает их на подбазисные коэффициенты у каскадов и на подбазисные показатели степеней («подбазисные» – это числа меньше базиса и поэтому уже не растущие при увеличении базиса операцией (а)).

Исследуя алгоритм гудстейнатора, мы видим, что счетчики постепенно становятся всё менее значимыми («сползают» с показателей на базисы), их число постепенно сокращается (хотя при «сползании» с показателя степени это число на время возрастает).

В конце концов исчезает последний счетчик, остается одни подбазисные числа. С этого момента пенроузовская операция (а) становится холостой – она уже ничего не делает. Операция же (б) продолжается, она уничтожает последнее подбазисное число, доходит до нуля и либо останавливается, если таковы условия «игры», либо уходит в отрицательную бесконечность.

Несмотря на первоначальную мощь операции (а), она КОНЕЧНА. Она заканчивается. А операция (б) БЕСКОНЕЧНА. Она не заканчивается. И ее бесконечность всегда «сильнее» конечной мощи операции (а).

В этом весь фокус «поразительной теоремы Гудстейна» (хитроумно скрытый от нас Пенроузом, чтобы побольше заморочить нам головы и создать впечатление каких-то необычных математических чудес).

Любое «доказательство» «теоремы Гудстейна» на самом деле всегда будет изучением работы «гудстейнатора» по тем принципам, которые я выше указал, – под какими бы одеждами это доказательство ни скрывалось.

Пенроуз говорит, что «*сам Гудстейн доказал свою теорему, прибегнув к разновидности метода, который называется «трансфинитной индукцией»*». На самом деле Гудстейн, конечно, просто изучал в подробностях алгоритм «гудстейнатора», и если это дело у математиков называется «трансфинитной индукцией», то это очень неудачное название. Словом «трансфинитный»

если это самый первый элемент списка). На списках можно строить древовидные структуры, когда от элемента одного списка начинается новый список или несколько списков.

[24] **МОИ:** Операционная система, созданная Валдисом Эгле и функционировавшая в Институте электроники и вычислительной техники Академии наук Латвийской ССР с 1976 по 1991 год.

обычно обозначают «то, что находится за бесконечностью» (например, «трансфинитные числа»). А у «гудстейнатора» никаких бесконечностей нет (кроме той отрицательной бесконечности, в которую уходит вычитание единицы после того, как она уничтожила все результаты наращивания базисов). Все счетчики гудстейнатора всегда остаются конечными (и тем самым конечны и все определяемые ими числа – какими бы громадными они ни были).

Пенроуз говорит, что «*при помощи одной лишь математической индукции доказать ее* (т.е. теорему Гудстейна) *невозможно*».

Это зависит от того, что называть «математической индукцией». Пенроуз ее тут же рядом определяет как «*сначала нужно проверить справедливость S(1), а затем показать, что, если верно S(n), то должно выполняться и S(n+1)*». Это определение индукции в духе математического формализма. А я с детства, с тех пор, как изучал математику в школе, не зная ничего о формалистах, привык считать, что «математическая индукция» означает: идти от n к n+1 и смотреть, что будет происходить при этом, – и что будет происходить, когда n неограниченно растет. И если ТАК понимать индукцию, то «теорему Гудстейна» мы (изучая алгоритм «гудстейнатора») доказываем именно математической индукцией. (Пенроуз – вслед за формалистами – обозначает словом «индукция» только те случаи, когда процесс идет монотонно в одном направлении, а я называю индукцией всякое пошаговое изучение процесса, даже в том случае, если процесс идет неравномерно и делает всякие там «петли» – как у «гудстейнатора»).

Но не в названиях, конечно, дело.

Однако далее Пенроуз провозглашает: «*теорема Гудстейна фактически является теоремой Гёделя для той самой процедуры, которую мы изучали в школе под названием математической индукции*». Вот как!

И чем же она «теорема Гёделя»? Оказывается, тем, что «математической индукцией» (понимаемой в узком формалистском смысле) теорему Гудстейна невозможно доказать, а она всё-таки справедлива!

По-моему, здесь, как редко где, раскрываются все изъяны пенроузовского мировоззрения.

«Теорема Гудстейна» – не что иное, как вывод о результатах действия некоторой программы (для мозгового или для промышленного компьютера), названной здесь «гудстейнатором». Программа эта – ну такая... – средней сложности (примерно как драйвер какого-нибудь устройства в Диспосе. Эта программа почти не выполнялась реально, но изучалась (Гудстейном, Пенроузом и нами) со стороны. Формалисты при их (урезанном) понимании индукции не могут теорему Гудстейна доказать индукцией (то есть, не могут разобраться в работе гудстейнатора). Поэтому они объявляют теорему Гудстейна «теоремой Гёделя» для математической индукции. (Очень умно! – великая наука, ничего не скажешь – хе-хе-хе!)

А Пенроуз пишет: «*метод* (доказательства теоремы Гудстейна) *сводится к систематизации интуитивных ощущений, которые возникают в процессе знакомства с «причиной», по которой теорема Гудстейна и в самом деле верна*». На самом деле мы убедились в правильности «теоремы Гудстейна», изучив алгоритм программы «гудстейнатора» и получив таким путем о нем определенные знания. И вот эти знания Пенроуз называет «интуитивными ощущениями». (Читатель! запомните, ЧТО он называет «математической интуицией»! – здесь это прекрасно видно!).

И далее Пенроуз пишет: «*Тем самым, в формировании нашего сознания с необходимостью есть элементы, которые не могут быть получены из какого бы то ни было набора вычислительных инструкций*».

Хорошо, посмотрим, какие «вычислительные инструкции» были необходимы Гудстейну, чтобы сформулировать, доказать и в 1944 году опубликовать свою теорему. А потом – как эти самые «инструкции» может выполнить Математический Интеллектуальный Киберкомплекс (МИК, *alias* Мик), оклеветанный Пенроузом в §3.23 книги «Тени разума» {PENRS2}.

В самых крупных блоках «верхнего уровня» аппарат самопрограммирования Гудстейна должен был создать (мозговые) программы для того, чтобы:

1) вообще вынести на рассмотрение задачу, в которой разбирается рост базисов и вычитание единицы;

2) составить мозговую программу собственно «гудстейнатора», т.е. определить, как всё будет происходить, какие будут строиться структуры данных (рис. VE1);

3) определить ключевые моменты для анализа алгоритма гудстейнатора (в первую очередь: что самым ключевым является момент, когда подбазисное число = 0, а проводится вычитание единицы);

4) установить закономерности протекания ключевых моментов алгоритма;
5) сформулировать «теорему Гудстейна»;
6) изложить всё это на бумаге;
7) отнести в редакцию журнала «*Journal of Symbolic Logic*».

Каждый из этих программных блоков выполняет (и у Гудстейна, и у Мика) довольно сложную работу (уж намного сложнее простого гудстейнатора). Причем эти программы не даны им в готовом виде «от рождения»; они должны быть ими самими составлены перед их выполнением.

Но в принципе мне понятно, как должны быть устроены программы каждого из этих блоков, а также то, как эти программы должны создаваться (предшествующими программами). Подробное рассмотрение этих вещей заняло бы очень много места и было бы, скорее всего, малопонятно читателю.

Однако здесь нам важно только одно: какое отношение к построению и работе программ этих семи блоков имеет тот факт, что при помощи (ограниченно понимаемой) математической индукции формалисты не могут доказать теорему Гудстейна? Каким образом этот факт доказывает невозможность существования программ блоков (1–7)?!

Задав этот вопрос читателю, я возвращаю слово Пенроузу:

* * *

Возможные «узкие места» в этом рассуждении сводятся к следующему. Наша способность (математического) познания может быть результатом вычислительной процедуры или непознаваемой из-за своей сложности; или не непознаваемой, но правильность которой, однако, не может быть установлена; или же ошибочной, хотя почти правильной.[25] Говоря об этом, мы должны прежде всего установить, откуда могут возникать подобные вычислимые процедуры. В книге *Тени разума* я достаточно подробно рассмотрел все такие «узкие места», и я хотел бы порекомендовать эту книгу (равно как и статью *Beyond the Doubting of a Shadow* в журнале *Psyche*[26] всем читателям, кому интересно было бы ближе познакомиться с настоящим предметом.

Если мы согласимся с тем, что в нашей способности познавать – а следовательно, и в нашей сознательной деятельности в целом – есть нечто, выходящее за пределы чисто алгоритмических действий, то следующим шагом мы должны попытаться выяснить, в каких из наших физических действий может проявляться «существенно неалгоритмическое поведение». (При этом мы негласно предполагаем, что изучение именно «физического действия» определенного вида поможет нам разгадать тайну происхождения сознания.) Я пытаюсь доказать, что таким «неалгоритмическим действиям» нельзя найти место в рамках общепринятых сегодня физических теорий. А значит, мы должны искать соответствующее место, где в научной картине существует серьезный пробел. И я утверждаю, что это «белое пятно» лежит где-то на границе между «субмикроскопическим» миром, в котором правит квантовая механика, и непосредственно воспринимаемым нами макромиром, подчиняющимся законам классической физики.

Здесь необходимо сделать важное замечание. Термин «невычислимый» относится к некоторому классу математических действий, про которые известно – то есть доказано математически, – что они не поддаются вычислениям. И одна из задач данной книги заключается в том, чтобы познакомить читателя с этим вопросом. Невычислимые процессы могут быть полностью детерминистскими. Эта особенность является диаметрально противоположной по отношению к свойству полной случайности, которое характерно для современной интерпретации квантовой механики и возникает при увеличении микромасштабных квантовых эффектов до классического уровня – R-процедуре в моей терминологии в этой книге. Я считаю, что необходима новая теория, которая позволит постичь смысл «реальности», принадлежащей сфере действия R-процедуры, которая сегодня используется в квантовой механике; и, как мне кажется, именно в этой неоткрытой пока новой теории мы найдем требуемый элемент невычислимости.

Кроме того, я смею утверждать, что эта недостающая теория является одновременно и искомым звеном между квантовой механикой и общей теорией относительности Эйнштейна. Для этой единой теории в физике применяется название «квантовая гравитация». Однако, большинство работающих в этой области ученых полагают, что объединение двух величайших теорий двадцатого века не затронет законов квантовой механики, в то время как общая теория относи-

[25] **В.Э.:** И это относится к программам блоков (1–7)? Я вынужден просто пожать плечами. Это же какая-то несуразица.

[26] См. сноску 2 на с. 12.

тельности должна претерпеть изменения. Я придерживаюсь иной точки зрения, поскольку считаю, что методы квантовой теории (в частности, R-процедура) тоже должны существенно измениться. В этой книге я использовал термин «правильная квантовая теория гравитации» (или «ПКТГ»), чтобы обозначить возможный результат такого объединения – хотя это и не будет теорией квантовой гравитации в обычном смысле (и, вероятно, «ПКТГ» тоже не очень удачный термин, который может ввести кого-то в заблуждение).

Хотя такой теории до сих пор не существует, это вряд ли может помешать нам оценить уровень, на котором она становится применимой. В книге я использовал для этих целей «одногравитонный критерий». Но несколько лет спустя я был вынужден изменить свои взгляды и, как мне кажется, найти более адекватный подход, изложенный в книге *Тени разума*. Этот подход близок к реальности не только «физически» (чему нашлось дополнительное подтверждение, которое я привел в одной[27] из своих статей), но и с практической точки зрения, что подтолкнуло нас к дальнейшим теоретическим изысканиям. На самом деле, сейчас уже разработан ряд физических экспериментов, которые, надеюсь, можно будет осуществить в ближайшие несколько лет.[28]

Но даже если всё перечисленное окажется справедливым и мои умозаключения подтвердятся, это не поможет нам отыскать «местоположение сознания». Вероятно, один из недостатков этой книги заключается в том, что к моменту завершения работы над ней я так и не знал, в каком месте мозга может происходить «крупномасштабная квантовая когерентность», которая необходима для использования приведенных выше идей. С другой стороны, к достоинствам книги следует отнести то, что она вызвала живой интерес в самых широких научных кругах, представители которых могут внести ценный вклад в исследования этого вопроса. Одним из таких ученых оказался Стюарт Хамерофф, который познакомил меня с цитоскелетом клетки и входящими в него микроканальцами – структурами, о которых я, к сожалению, не имел ни малейшего представления! Он также изложил мне свои оригинальные идеи по поводу возможной роли микроканальцев в нейронах мозга для феномена сознания – что позволило мне предположить, что они-то и являются скорее всего тем местом, где может происходить крупномасштабная квантовая когерентность, на которую я опирался в своих рассуждениях. Конечно же, эта информация достигла меня уже слишком поздно, чтобы я мог включить ее в настоящее издание; но ее изложение можно найти в книге *Тени разума* и последующих статьях, написанных преимущественно в соавторстве со Стюартом Хамероффом.[29]

Кроме последних достижений, упомянутых в этом новом вступлении, можно сказать, что все основные идеи книги *Новый ум короля* сохранились в том же виде, что и десять лет назад. Я надеюсь, что читатель, познакомившись с изложенными здесь мыслями, получит неподдельное удовольствие и почувствует желание самостоятельно продолжить изучение этих вопросов.

Роджер Пенроуз
Сентябрь 1998

---

[27] *On the gravity role in quantum state reduction*, General Relativity and Gravitation, 28 (1996), 581–600.

[28] См. Penrose, R., *Quantum computation, enanglement and state reduction*, Phil. Trans. Royal Soc. London, A356 (1998), 1927–39; и Moroz, I., Penrose, R. and Tod, K.P., *Spherically symmetric solutions of the Schrödinger–Newton equations*, Classical and Quantum Gravity, 15 (1998).

[29] Hameroff, S.R. and Penrose, R., *Conscious events as orchestrated space-time selections*, J. Consciousness Studies, 3 (1996), 36–63. Hameroff, S.R. and Penrose, R., *Orchestrated reduction of quantum coherence in brain microtubules – a model for consciousness*; в сборнике Towards a science of consciousness: contributions from the 1994 Tucson Conference (ed. S. Hameroff, A. Kazniak and A. Scott), MIT Press 1996. Hameroff, S.R., *Fundamental Geometry: the Penrose–Hameroff “Orch OR” model of consciousness* в сборнике The geometric universe, science, geometry and the work of Roger Penrose (ed. S.A. Huggett, L.J. Mason, K.P. Tod, S.T. Tsou and N.M.J. Woodhouse), Oxford University Press, 1998.

Пролог

На церемонию запуска нового компьютера *Ультроник* в Большой аудитории собралась огромная толпа. Президент Полло только что закончил свое вступительное слово. Он рад, что наконец отделался – подобные мероприятия ему не по вкусу, а в компьютерах ему интересно лишь одно: эта новая штуковина позволит ему сэкономить кучу времени. Разработчики уверяли его, что помимо всего прочего, *Ультроник* будет способен отвечать за принятие решений в государственных делах, которые всегда докучали президенту. И неплохо бы, чтобы это оказалось правдой – учитывая то, сколько за этот компьютер заплачено золота из казны! Президент уже предвкушал многочасовые игры в гольф на своем личном поле – одном из немногих оставшихся в его крохотной стране островков зелени.

Адаму лестно находиться среди приглашенных на церемонию открытия. Он сидит в третьем ряду; через два ряда впереди него сидит его мать, главный технолог разработки *Ультроник*. Вышло так, что и отец его тоже находится здесь: он пришел без приглашения и сидит сейчас в самом конце зала, окруженный со всех сторон охранниками – и всё потому, что в последний момент отец решил взорвать компьютер. Он сам поручил себе это задание как доморощенный лидер маленькой группы маргинальных активистов, именующей себя Высший Совет Психического Самосознания. Конечно, всю его взрывчатку тут же обнаружили установленные в изобилии электронные и химические датчики, и в качестве наиболее приятной части предстоящего наказания ему довелось стать невольным свидетелем церемонии запуска.

Адам не испытывал особых чувств ни к одному из своих родителей. Быть может, в таких чувствах у него и не было необходимости: все тринадцать лет своей жизни он рос в атмосфере материальной роскоши, обусловленной в основном возможностями компьютеров. Любое свое желание он мог удовлетворить простым нажатием на кнопку мыши – будь то потребность в еде, питье, компании или развлечениях, а если нужно, то и в знаниях – и всегда это сопровождалось прекрасными цветными иллюстрациями на графических мониторах. Всё было возможно благодаря положению, которое занимала мать Адама.

И вот Главный конструктор проекта уже заканчивает свой доклад: «...более 10^{17} логических ячеек. Это больше, чем суммарное число нейронов у всех живущих в нашей стране! Уровень интеллекта невообразимо высок. Но, к счастью, нам и не нужно ничего воображать – через минуту у каждого будет возможность убедиться в этом собственными глазами! Я попрошу уважаемую первую леди нашей великой страны, мадам Изабеллу Полло, включить рубильник питания нашего фантастического компьютера *Ультроник*!»

Супруга президента подается вперед. Немного нервничая и чуть колеблясь, она поворачивает рубильник. Небольшой шорох, еле ощутимое мерцание индикаторов – и вот, 10^{17} логических ячеек активированы! Все замерли в ожидании, не совсем представляя, чего собственно и ожидать. «Итак, найдется в этой аудитории желающий инициировать нашу новую компьютерную систему *Ультроник*, задав ей первый вопрос?» – обращается к залу Главный конструктор. Всеобщая растерянность. Никто не решается, дабы не оказаться глупцом при таком скоплении народа – и перед новым Вездесущим Разумом. Тишина. «Ну что же вы, наверняка кто-то хочет задать вопрос!» – не сдается Главный конструктор. Все в смятении, как будто чувствуя присутствие нового всемогущего разума. Лишь Адам хладнокровен. Он окружен компьютерами с самого рождения. Он почти чувствует, что значит быть компьютером. Или, по крайней мере, ему так кажется. Во всяком случае, он заинтригован. Адам поднимает руку. «Ну вот, – говорит Главный конструктор, – парнишка в третьем ряду. У тебя есть вопрос к нашему... гм... нашему новому другу?»[30]

---

[30] **В.Э.:** А в Эпилоге Пенроуз это заканчивает так: «*Эпилог. «...СЕБЯ ЧУВСТВУЕШЬ? О... весьма интересный вопрос, мой мальчик... э-э... я и сам хотел бы знать ответ», – сказал Главный конструктор. – «Давайте посмотрим, что может сказать наш друг об этом... странно... э-э... Ультроник говорит, что он не понимает, что... он не может даже понять, что ты имеешь в виду!» Отдельные смешки в аудитории переросли в громовой хохот. Адам чувствовал себя крайне неловко. Они могли отреагировать как угодно, но только не смеяться.*» То есть, предполагается, что компьютер с искусственным интеллектом не будет себя чувствовать... (Ха-ха!)

Глава 1. Может ли компьютер обладать разумом?

§1.1. Введение

На протяжении нескольких предыдущих десятилетий компьютерные технологии развивались семимильными шагами. Более того, нет никаких сомнений в том, что и будущее сулит нам новые грандиозные успехи в повышении быстродействия и объема памяти, а также новые конструктивные решения компьютерной логики. Сегодняшние компьютеры завтра покажутся нам такими же медленными и примитивными, как механические калькуляторы прошлого. В таком стремительном развитии есть что-то почти пугающее. Уже сейчас машины способны решать различные задачи, ранее являвшиеся исключительной прерогативой человеческого интеллекта. И решать их со скоростью и точностью, во много раз превосходящими человеческие способности. Мы давно свыклись с существованием устройств, превосходящих наши физические возможности. И это не вызывает у нас внутреннего дискомфорта. Наоборот, нам более чем комфортно, когда автомобиль несет нас в пять раз быстрее, чем лучший в мире бегун. Или когда с помощью таких устройств мы копаем ямы или сносим непригодные конструкции – с эффективностью, которую не разовьет и отряд из нескольких дюжин добрых молодцев. Еще больше нам импонируют машины, с помощью которых у нас появляется возможность делать то, что нам ранее было попросту недоступно физически, например, подняться в небо и всего через несколько часов приземлиться на другом берегу океана. Эти машины не задевают нашего тщеславия. Но вот способность мыслить всегда была прерогативой человека. В конце концов, именно этой способности мы обязаны тому, что человеку удалось преодолеть его физические ограничения и встать в развитии на ступеньку выше над другими живыми существами. А если когда-нибудь машины превзойдут нас там, где, по нашему мнению, нам нет равных – не получится ли так, что мы отдадим пальму первенства своим же собственным творениям?

Можно ли считать, что механическое устройство в принципе способно мыслить, или даже испытывать определенные чувства? Этот вопрос не нов,[31] но с появлением современных компьютерных технологий он приобрел новое значение. Смысл вопроса глубоко философский. Что значит – думать или чувствовать? Что есть разум? Существует ли он объективно? И если да, то в какой степени он функционально зависим от физических структур, с которыми его ассоциируют? Может ли он существовать независимо от этих структур? Или он есть лишь продукт деятельности физической структуры определенного вида? В любом случае – должны ли подходящие структуры быть обязательно биологическими (мозг) или, возможно, этими структурами могут быть и электронные устройства? Подчиняется ли разум законам физики? И вообще, что такое законы физики?

Вот часть проблем, которые я попытаюсь затронуть в этой книге. Просить дать определенный ответ на такие глобальные вопросы – это, конечно, было бы слишком. Я не способен дать такой ответ, да и никто не способен[32] – хотя некоторые, возможно, попытались бы вас обескура-

---

[31] См., например, работы Гарднера [1958], Грэгори [1981] и содержащиеся там ссылки. **В.Э.:** Список литературы в {PENRO5 = МОИ № 16}.

[32] **В.Э.:** Это вопрос гносеологический: как вообще человек может что-то достоверно знать? Меня, например, с детства терзают различные сомнения: «А была ли вообще Вторая мировая война?», «А жил ли вообще мой прадед?», «А появляются ли вообще дети в результате полового акта?» – я и сейчас как-то не очень уверен во всех этих вопросах и тысячах других вещей (которые окружающие меня люди, судя по их поведению, считают несомненными). А вдруг всё это – обман? Так что с этой точки зрения я не уверен абсолютно ни в чем (пожалуй, даже в том, что я существую – вопреки Декарту с его «*Cogito, ergo sum*»). Но, с другой стороны, невозможно жить в таком мире – до такой степени неопределенном. Для выработки практического поведения приходится принимать по каждому вопросу наиболее вероятные гипотезы. Для нахождения таких наиболее вероятных предположений существует целая система критериев (среди которых наиболее ярким алмазом сверкает, пожалуй, «лезвие Оккама»). Для практического поведения следует найти это «наиболее вероятное предположение» и потом держаться так, как будто ты и вправду считаешь его истинным, не тратя больше времени на сомнения и не высказывая никаких колебаний. Таковы решения, например, по вопросу о Боге: по всем критериям наиболее вероятное предположение состоит в

жить своими догадками. Мои собственные догадки играют большую роль в последующем изложении, но я постараюсь очень внимательно подчеркивать, где кончается строгий научный анализ и начинаются догадки, а также то, чем мои соображения мотивированы. Я не пытаюсь угадать правильные ответы: моя главная задача куда скромнее. Цель этой книги – поднять ряд, по-видимому, новых вопросов о взаимосвязи структуры физических законов, естества математики и разумного мышления, а также представить точку зрения, отличную от тех, которые я когда-либо встречал. Я не могу описать эту точку зрения в двух словах – вот одно из объяснений того, почему я решил написать книгу такого объема. Но если суммировать кратко (хотя краткость вполне может ввести читателя в заблуждение), моя позиция основана на осознании того, что именно наше недостаточное понимание фундаментальных физических законов препятствует построению концепции «разума» в физических и логических терминах. Я не утверждаю, что мы никогда не познаем физические законы в достаточной для этого степени. Наоборот, одна из задач книги – попытаться дать стимул дальнейшим исследованиям в наиболее перспективных в данном отношении направлениях, и попробовать пояснить достаточно определенные (и, вероятно, свежие) соображения о месте, которое могло бы занимать понятие «разума» в известной нам физической науке.

Сразу отмечу, что моя точка зрения не является общепринятой среди физиков. Поэтому маловероятно, что в настоящее время она получит признание ученых-компьютерщиков или психологов. Любой физик скажет вам, что фундаментальные законы, действующие на масштабах, характерных для человеческого мозга, прекрасно известны. Хотя никто не отрицает, что в наших знаниях физики как таковой многого недостает. Мы, например, не знаем ни основных законов, которые определяют значения масс субатомных частиц, ни законов, определяющих силу взаимодействия между этими частицами. Мы не знаем, как добиться полного согласования квантовой теории и специальной теории относительности Эйнштейна – не говоря уже о том, как построить теорию квантовой гравитации, в рамках которой удалось бы согласовать квантовую теорию и <u>общую</u> теорию относительности. Вследствие этого мы не способны понять природу пространства на чрезвычайно малых расстояниях порядка 1/100˙000˙000˙000˙000˙000˙000 размеров известных фундаментальных частиц, хотя и считается, что на бóльших расстояниях наши представления являются адекватными. Мы не знаем, является ли вселенная как единое целое конечной или бесконечной в пространственных или во временном измерениях, хотя подобные неопределенности, по-видимому, совершенно несущественны для физики важных для человека явлений. Мы не представляем себе, какие физические законы работают в сердцевине черных дыр и какие законы действовали в момент Большого взрыва при рождении самой нашей вселенной. Все перечисленные проблемы, однако, кажутся нам невообразимо далекими от шкалы явлений «повседневной» жизни (или чуть меньшей шкалы), от масштабов, характерных для жизнедеятельности человеческого мозга. И эти проблемы действительно невообразимо далеки! Тем не менее, я утверждаю, что в нашем понимании физического мира есть брешь <u>именно на том</u> уровне, который может иметь непосредственное отношение к работе человеческого мозга и сознанию. Эта брешь – прямо у нас под носом (или, скорее, за ним)! Однако большинство физиков даже не чувствуют ее – ниже я попытаюсь объяснить почему. Далее я приведу доводы в пользу того, что теории черных дыр и Большого взрыва на самом деле <u>имеют</u> определенное отношение к рассматриваемым вопросам!

Ниже я постараюсь убедить читателя в силе рассуждений, лежащих в основе предлагаемой мною точки зрения. Но чтобы понять ее, потребуется изрядно потрудиться. Нам понадобится совершить путешествие в довольно странные области (кажущиеся, возможно, не имеющими отношения к делу) и заглянуть во многие сферы научной деятельности. Будет необходимо подробно изучить структуру, основы и парадоксы квантовой теории, основные положения специальной и общей теории относительности, теории черных дыр, Большого взрыва, второго закона термодинамики, максвелловской теории электромагнитных явлений, а также основы механики Ньютона. При попытке понять природу и работу сознания в игру немедленно войдут

том, что Его нет и что Он не влияет на нашу жизнь и вообще на наше существование – и нечего тут больше ломать голову! Аналогично наиболее вероятное предположение: – что Земля действительно имеет шарообразную форму, что она и вправду вертится вокруг Солнца и что нас в этом не обманывают. Если по таким же критериям отобрать «наиболее вероятное предположение» о сущности человеческой психики и интеллекта, то оно заключается в том, что всё это работа биологической системы обработки информации, и не требуется там никаких других объяснений, ни «божественной», ни «квантовой» природы. И в <u>таком</u> смысле я <u>способен</u> дать ответы на названные Пенроузом «глобальные вопросы».

также философия и психология. Имея перед собой компьютерные модели, мы, конечно, не обойдемся и без экскурса в нейрофизиологию живого мозга. Нам понадобится также некоторое представление о статусе искусственного интеллекта. Потребуется разобраться, что такое машина Тьюринга, понять смысл вычислимости, теоремы Гёделя и теории сложности. Кроме того, нам придется окунуться в дебри оснований математики и даже обсудить вопрос о самой природе физической реальности.

И если после всего этого читатель останется скептически настроен к наиболее необычным из моих аргументов, то мне, по крайней мере, хочется верить, что он вынесет нечто действительно ценное из этого изматывающего, но (я надеюсь) увлекательного путешествия.

§1.2. Тест Тьюринга

Представьте себе, что появилась новая модель компьютера, объем памяти и число логических ячеек которого больше, чем у человеческого мозга. Представьте далее, что такие компьютеры грамотно запрограммированы и в них введено огромное количество необходимых данных. Производители убеждают вас, что эти устройства могут на самом деле мыслить, и, возможно, утверждают, что подобные компьютеры в действительности являются разумными. Или они идут еще дальше и заявляют, что эти машины могут чувствовать – чувствовать боль, радость, сострадание, гордость и т.п., и что они на самом деле понимают, что делают. То есть, как будто бы утверждается, что машины обладают сознанием.

Как нам понять, можно ли верить производителям? Когда мы покупаем устройство, мы, как правило, судим о его качестве лишь по полезным для нас функциям. Если устройство работает по назначению, оно нас устраивает. Если нет – его ремонтируют или меняют на новое. Чтобы проверить справедливость утверждений производителей о наличии человеческих качеств у данного устройства, мы должны, в соответствии с указанным критерием, всего лишь потребовать от устройства поведения, повторяющего поведение человека в отношении данных качеств. Если устройство поведет себя удовлетворительно, к производителям нет претензий, и компьютер не требует возврата для ремонта или замены.

Такая схема дает существенно операционалистский подход к рассмотрению подобных вопросов. Операционалист скажет вам, что компьютер мыслит, если компьютер ведет себя точно так же, как и человек в момент раздумий. Примем, для начала, эту операционалистскую точку зрения. Естественно, от компьютера здесь не требуется расхаживать по комнате, подобно тому, как мог бы вести себя размышляющий о чем-то человек. Еще меньше мы озабочены тем, чтобы компьютер был внешне похож на человека или напоминал на ощупь человеческое тело: эти качества не имеют отношения к назначению компьютера. То, что нас действительно интересует – его способность выдавать схожие с человеческими ответы на любой вопрос, какой нам заблагорассудится ему задать. И мы примем, что компьютер на самом деле думает (чувствует, понимает и т.д.), если его манера отвечать на наши вопросы будет неотличима от человеческой.

Этот подход очень горячо отстаивался в знаменитой статье Алана Тьюринга [1950] *Вычислительные машины и интеллект*, появившейся в 1950 году в философском журнале *Mind*. (Фамилию Тьюринг мы еще встретим позже.) В этой статье впервые была предложена идея того, что сейчас называют тестом Тьюринга. Тест предназначался для ответа на вопрос о том, можно ли резонно утверждать, что машина думает. Пусть утверждается, что некоторый компьютер (подобный тому, который продают производители из описания выше) в действительности думает. Для проведения теста Тьюринга компьютер вместе с человеком-добровольцем скрывают от глаз (проницательной) опрашивающей.[33] Опрашивающая[34] должна попытаться определить, где компьютер, а где человек, задавая им двоим пробные вопросы. Вопросы, а еще важнее – ответы, которые она получает, передаются в безличной форме, например, печатаются на клавиатуре и высвечиваются на экране. Единственная информация, которой будет располагать опрашивающая

---

[33] Неизбежная проблема в изложении подобного рода – дилемма между «он» и «она» в контексте, не подразумевающем никакого указания на пол. Ссылаясь на некоторый абстрактный персонаж, я далее буду использовать местоимение «он», имея в виду «он или она» – это, по-моему, является обычной практикой. Однако в данном случае я отдаю предпочтение женщине в роли опрашивающего, и, надеюсь, меня простят за этот единственный пример явной «половой дискриминации». Я думаю, что при определении истинно человеческих качеств женщина окажется чувствительней своего мужского двойника!

[34] **В.Э.:** Точка зрения, что женщина лучше, чем мужчина почувствует, где настоящая человеческая природа и где обман, – ошибочна. На самом деле именно женщины более доверчивы, чем мужчины, их (в среднем) легче обмануть, и именно они чаще становятся жертвами всякого обмана и мошенничества.

– это то, что она сама сможет выяснить в процессе такого сеанса вопросов и ответов. Опрашиваемый человек честно отвечает на все вопросы, пытаясь убедить женщину, что он и есть живое существо; компьютер, однако, запрограммирован таким образом, чтобы обмануть опрашивающую и убедить ее в том, что человек на самом деле он. Если в серии подобных тестов опрашивающая окажется неспособной «вычислить» компьютер никаким последовательным образом, то считается, что компьютер (или компьютерная программа, программист, разработчик и т.д.) прошел данный тест.

Можно возразить, что тест на самом деле не очень-то честный по отношению к компьютеру. Если бы роли человека и машины поменялись, и человеку нужно было бы прикидываться компьютером, определить «кто есть кто» не составило бы никакого труда: опрашивающей лишь стоило бы задать какой-нибудь очень сложный арифметический пример. Хороший компьютер тут же выдал бы правильный ответ,[35] а человек оказался бы в замешательстве. (Здесь, однако, следует проявить осторожность. Среди людей известны «вычислительные дарования», способные в уме решать весьма нетривиальные счетные задачи с безошибочной точностью и без всяких видимых усилий. Например, сын неграмотного крестьянина Иоганн Мартин Захария Дазе,[36] живший в Германии с 1824 по 1861 год, в уме перемножал любые два восьмизначных числа менее чем за минуту, а за шесть минут он перемножал два двадцатизначных числа! Такие способности не мудрено принять за результат работы компьютера.[37] Более поздний пример (1950-е годы) – столь же исключительные вычислительные способности Александра Айткена, профессора Эдинбургского университета. Нужно, чтобы арифметическое задание опрашивающей было гораздо сложнее – например, перемножить два тридцатизначных числа за две секунды. Хороший современный компьютер запросто справится с таким упражнением.)

Итак, часть задачи программистов состояла бы в том, чтобы в некоторых вещах компьютер казался глупее, чем он есть на самом деле. Если опрашивающая задает сложный арифметический пример, подобный приведенному выше, компьютер должен притвориться, что не в силах на него ответить – иначе его немедленно изобличат![38] Я, правда, не думаю, что задача сделать компьютер глупее в указанном смысле является серьезной проблемой для программистов компьютеров. Главная сложность – научить компьютер отвечать на простейшие вопросы на проверку «здравого смысла», с которыми у человека вообще не будет проблем!

У конкретных вопросов такого типа есть, однако, одно слабое место. Каков бы ни был вопрос, легко придумать способ заранее научить компьютер отвечать на <u>данный</u> вопрос точно так же, как на него ответил бы человек. И тем не менее, недостаток понимания компьютером сути весьма вероятно обозначится при <u>продолжительном</u> опросе, особенно если вопросы носят нестандартный характер и требуют настоящего осмысления. Искусство опрашивающей должно включать как умение изобрести оригинальные вопросы, так и умение дополнить их позже другими вопросами на понимание таким образом, чтобы выяснить, действительно ли вопросы были усвоены. Кроме того, она может периодически подбрасывать бессмысленные вопросы (сможет ли компьютер их распознать?), или вставлять один-другой с виду бессмысленный, но на деле все-таки имеющий смысл вопрос. Например, она может спросить: «Я слышала, что сегодня утром носорог летел вверх по Миссисипи на розовом воздушном шаре. Что Вы об этом думаете?» (Тут можно живо представить себе, как лоб компьютера покрывается капельками

[35] **В.Э.:** Глубокое заблуждение! Всё будет зависеть от той программы, которая будет принимать заданный вопрос. Она может и вообще не уметь работать с числами. Или считать медленно – «на пальцах». Но если задача программиста состояла в том, чтобы показать, что это компьютер, то он, конечно, в своей программе обратится к встроенным в компьютер арифметическим операциям, которые выполняются быстро. Однако, если надо, наоборот, притвориться человеком, то ничего тормозить (как думает Пенроуз) на самом деле не надо: достаточно просто не обращаться к встроенным арифметическим операциям.

[36] См., например, работу Резникова и Уэллса [1984]. Ознакомиться с классическими обзорными работами по «чудесам» вычислений можно у Рауза Болла [1982] и Смита [1983].

[37] **В.Э.:** Они и есть результат работы компьютера (биологического). Весь вопрос в том, какие программы этот (самопрограммирующийся) компьютер способен сгенерировать для решения данных ему задач. У большинства людей эти генераторы создают лишь очень неэффективные программы для арифметической работы. Но вот, изредка случается, что генераторы самопрограммирования создают и довольно эффективные программы – как у Иоганна Дазе или Александра Айткена. В принципе, конечно, мозг как компьютер мог бы выполнить чрезвычайно эффективные арифметические программы, но этому препятствует самопрограммирование – такие программы трудно создать этим путем.

[38] **В.Э.:** Эта фраза свидетельствует, что Пенроуз не очень хорошо разбирается в компьютерах и плохо представляет, как они работают.

холодного пота – если выбрать наименее подходящую метафору.) Он может оказаться начеку и ответить: «Пожалуй, это звучит странно». Что ж, пока неплохо. Женщина: «Правда? Мой дядя как-то проделал это, причем туда и обратно, только на сероватом с полосками. Чего же тут странного?» Ясно, что без понимания компьютер скоро будет разоблачен.[39] Отвечая на первый вопрос, он может даже ляпнуть: «Носороги не летают», – если в банках памяти удачно всплывет информация о том, что у них нет крыльев. Или ответить на второй вопрос, что носороги не бывают полосатыми.[40] А дальше женщина может, например, подсунуть совершенно бессмысленный вопрос, заменив отдельные слова: «под Миссисипи», или «внутри розового воздушного шара» и т.п., и выяснить, хватит ли у компьютера здравого смысла, чтобы обнаружить существенное различие!

Оставим на время в стороне вопрос о том, возможно ли (а если да, то когда станет возможно) создание компьютера, который пройдет тест Тьюринга. Предположим вместо этого – исключительно для того, чтобы обсудить проблему – что такие машины уже созданы. Возникает резонный вопрос, должен ли прошедший тест компьютер непременно быть признан мыслящим, чувствующим, понимающим и т.д.? Этот вопрос мы рассмотрим очень скоро, а пока обсудим некоторые связанные с ним аспекты. Например такой: если производители честны во всех своих самых смелых заявлениях и их устройство есть мыслящее, чувствующее, понимающее, сознательное существо, то покупка устройства возлагает на нас моральную ответственность. Так непременно должно быть, если производителям можно верить. Использовать такой компьютер для наших нужд и не учитывать его переживаний было бы предосудительно. С моральной точки зрения такое использование – это то же, что и жестокое обращение с рабом. Прежде всего, мы были бы должны избегать причинить компьютеру боль, которую, по утверждениям производителей, он способен чувствовать. Выключение компьютера, возможная его продажа после того, как компьютер к нам привык, были бы сопряжены для нас с моральными проблемами. Таких проблем возникло бы великое множество, и они были бы того же сорта, что и проблемы, которые возникают у нас в отношениях с другими людьми и живыми существами. Всё это стало бы для нас вопросом первостепенной важности. И крайне важной для нас (да и для административных органов!) стала бы уверенность в том, что реклама производителей типа «Каждое мыслящее устройство прошло тщательное тестирование по Тьюрингу группой наших экспертов!» действительно является правдой.[41]

Несмотря на очевидную абсурдность некоторых аспектов рассматриваемого вопроса (в частности, моральных), мне кажутся достаточно обоснованными доводы в пользу того, что успешно пройденный тест Тьюринга есть указание на присутствие мысли, интеллекта, понимания или сознания. В самом деле, на чем еще могут основываться наши убеждения в присутствии этих качеств у других людей, кроме как на беседе с ними? Строго говоря, другие критерии тоже существуют: выражение лица человека, движения его тела и, вообще, его действия могут оказать на нас весьма сильное влияние. Не будет ничего сверхъестественного, если (возможно, в недалеком будущем) появится робот, который сможет удачно имитировать

---

[39] **В.Э.:** Конечно, ясно, что без понимания будет разоблачен, и мне даже не очень верится, что кто-то может думать иначе. Но вопрос состоит в том, что́ есть понимание, и можно ли его встроить в компьютер. И для того, кто понимает, что из себя представляет «понимание», очевидно, что его можно встроить в компьютер – нужно просто знать: КАК.

[40] **В.Э.:** Здесь какая-то путаница: сероватым с полосками же вроде был воздушный шар, а не носорог. Английский текст звучит так: «For example she might say, `I hear that a rhinoceros flew along the Mississippi in a pink balloon, this morning. What do you make of that?' (One can almost imagine the beads of cold sweat forming on the computer's brow – to use a most inappropriate metaphor!) It might guardedly reply, `That sounds rather ridiculous to me'. So far, so good. Interrogator: `Really? My uncle did it once – both ways – only it was off-white with stripes. What's so ridiculous about that?' It is easy to imagine that if it had no proper `understanding', a computer could soon be trapped into revealing itself. It might even blunder into `Rhinoceroses can't fly', its memory banks having helpfully come up with the fact that they have no wings, in answer to the first question, or `Rhinoceroses don't have stripes' in answer to the second.».

[41] **В.Э.:** Описанная здесь Пенроузом ситуация нереальна. Все эти вопросы возникнут уже задолго ДО того, как «производителем» будет «рекламироваться» мыслящее существо. Эти вопросы поднимутся еще на стадии проектирования разработки; уже тогда надо будет решать: зачем создается ИИ, с какой целью, с какими характеристиками, что с ним делать дальше и т.д. А к стадии «продажи» (если таковая вообще наступит, что очень маловероятно), всё уже будет выглядеть совсем иначе, чем в описании Пенроуза.

человеческую мимику и жесты. Тогда необходимость прятать робота и человека от опрашивающей отпадет, но критерии теста, которые будут у нее в распоряжении, останутся неизменными.

Лично я готов к тому, чтобы значительно упростить тест Тьюринга. Мне кажется, что требовать от компьютера идеального подражания человеку так, чтобы стать неотличимым от него в каких-то существенных вопросах, это требовать от компьютера больше, чем надо. Мне бы хватило, чтобы наша проницательная опрашивающая по ответам на свои вопросы просто убедилась, что имеет дело с сознательным разумом, пусть даже чужеродным. Вот то, что реально недостижимо во всех созданных на сей день компьютерных системах. Предвижу, однако, вероятность того, что после разоблачения компьютера у опрашивающей может возникнуть (возможно, подсознательное) нежелание приписать ему разумные качества даже тогда, когда она способна эти качества различить. Или наоборот, у нее может создаться впечатление «присутствия чужеродного разума», и она станет подыгрывать компьютеру, даже если «чужеродного разума» и нет. Поэтому исходный вариант теста Тьюринга гораздо предпочтительней в силу большей объективности, и ниже я обычно буду придерживаться той схемы. Присущая ей «несправедливость» по отношению к компьютеру, о которой говорилось выше (чтобы пройти тест, компьютер должен уметь всё, что и человек, а человек не обязан иметь способности компьютера), не смущает сторонников теста Тьюринга, считающих этот тест точным испытанием на способность мыслить, чувствовать и т.д. Во всяком случае, многие из сторонников теста придерживаются той точки зрения, что до того, как компьютер будет способен в действительности пройти тест, ждать осталось недолго – скажем, до 2010 года. (По прогнозам самого Тьюринга, 30%-ное успешное прохождение теста с опрашивающим «средних» способностей и всего с 5-минутным ограничением на продолжительность опроса могло бы быть реализовано к 2000 году.) Они уверены, что даже такая «предубежденность» не способна существенно отодвинуть эту дату!

Всё вышеизложенное становится важным, коль скоро ставится вопрос по сути: дает ли операционалистская схема приемлемый набор критериев, позволяющих судить о присутствии или отсутствии мыслительных способностей у объекта? По мнению некоторых, – нет, не дает. Имитация, какой бы искусной она ни была, не должна быть с необходимостью тем же, что и оригинал. Я занимаю в этом отношении скорее промежуточную позицию. Общий принцип, к которому я склоняюсь, состоит в том, что любая, даже самая искусная, имитация всегда должна быть обнаружима достаточно тщательным тестированием. Хотя, конечно, это скорее вопрос веры (или научного оптимизма), чем доказанный факт. Таким образом, в целом я готов принять тест Тьюринга как грубо адекватный в том контексте, в котором он определяется. То есть, если компьютер действительно окажется способен ответить на все заданные вопросы в точности так же, как на них ответил бы человек, и тем самым последовательно и честно[42] надуть нашу проницательную опрашивающую, то в отсутствие свидетельств об обратном моим предположением было бы то, что компьютер действительно думает, чувствует и т.д. Использование мною слов «свидетельство», «действительно» и «предположение» подразумевает, что когда я говорю о мышлении, чувствах, понимании, или, в частности, сознании, я не отношусь к этим понятиям как к элементам общепринятой лексики, а имею в виду конкретные и объективные «вещи», присутствие или отсутствие которых в физических телах есть то, в чем мы хотели бы удостовериться. И это я считаю ключевым моментом. Пытаясь уловить присутствие данных качеств, мы делаем предположения на основании всех доступных нам свидетельств. (В принципе, точно так же действует астроном, пытаясь вычислить массу далекой звезды.)

Какие же свидетельства об обратном принимать во внимание? Наперед заданные правила установить сложно. Однако, я сразу подчеркну: тот факт, что компьютер может состоять из транзисторов и проводов, а не нейронов и кровеносных сосудов, сам по себе не является аргументом, который я рассматривал бы как свидетельство об обратном. Меня не покидает мысль, что когда-нибудь будет построена удовлетворительная теория сознания – удовлетворительная в смысле логической последовательности и физической приемлемости, чудесной согласованности с другим физическим знанием. Ее предсказания будут в точности соотноситься

---

[42] Я придаю особое внимание тому, что я считаю честным прохождением теста Тьюринга. Я могу, например, представить ситуацию, в которой после длинной череды поражений компьютер будет запоминать все данные ранее человеком ответы и затем выдавать их в смеси с подходящими случайными добавками. Через какое-то время у нашей уставшей опрашивающей могут кончиться нетривиальные вопросы для беседы, и она окажется обманутой компьютером – тогда я назову это «мошенничеством» с его стороны!

с представлениями человека об уровне и условиях существования его собственного сознания, – и такая теория может оказаться в действительности плодотворной в разрешении проблемы предполагаемого наличия сознания у нашего компьютера.[43] Можно даже пофантазировать о «детекторе сознания», сконструированном по принципам такой теории – абсолютно надежном в случае человека, но дающем расходящиеся с тестом Тьюринга результаты в случае компьютера. Интерпретация результатов тестов Тьюринга тогда потребует особой осторожности. По моему мнению, отношение к вопросу о пригодности теста Тьюринга отчасти зависит от предположений о том, как будет развиваться наука и техника. Ниже нам еще придется вернуться к некоторым из этих рассуждений.

* * *

2010.08.30 12:09 понедельник

<u>**В.Э.:**</u> Здесь я опять сделаю большую вставку в текст Пенроуза. В 1999 году вышла моя книжка «Tur tālumā, kur ziemas nepazīst» с изложением по-латышски основ Веданской теории. Одним из тех, кто ее купил в магазине, был хабилитированный доктор ф.-м. наук профессор Юрис Тамберг;[44] вскоре он позвонил мне, мы встретились, он написал рецензию на мою книжку (единственный благожелательный отзыв о Веданской теории, который я получил за 32 года ее существования); я, в свою очередь, тоже написал ответ на его рецензию и на одну его статью, где тоже рассматривались эти вопросы, – он сам мне ее дал и предложил ответить.

Как раз профессор Тамберг и указал мне на книгу Пенроуза[45], и по его «наводке» я ее нашел и прочитал. В своих ответах профессору Тамбергу среди многих других вопросов я касался и книги Пенроуза. И вот, теперь я вставлю в этот том два фрагмента из тех ответов – написанных более десяти лет назад по-латышски, но переведенных на русский язык сейчас. Один фрагмент я вставляю здесь, после пенроузовской главы о тесте Тьюринга, а второй дальше – после пенроузовской главы о «Китайской комнате».

Итак, первый фрагмент из моего ответа[46] профессору Тамбергу (аббревиатурой «**В.Э.:**» отмечены подстрочные примечания, добавленные при русском переводе текста, а примечания без этой аббревиатуры имелись уже в латышском тексте):

(...)

§26. Китайская комната

2000.05.18 15:22 четверг

.310. Далее Вы пишете так:

> .311. «Всё же надо отметить, что ряд серьезных исследователей очень критически настроены к обобщениям такого рода, подчеркивая существенное различие между «мышлением» компьютера и человека. Среди тех выдающихся ученых, которые считают, что между деятельностью человеческого мозга и деятельностью компьютера имеются очень существенные различия, в первую очередь хотел бы упомянуть Р. Пенроуза, анализирующего эти проблемы в своей книге. Согласно Р. Пенроузу уже анализ некоторых самых простых примеров («тест Тьюринга» и «китайская комната») показывает, что сравнение деятельности человеческого мозга и компьютера само по себе является очень трудной задачей, для которой невозможны упрощенные решения. Например, в случае теста «китайской комнаты» Р. Пенроуз показывает, что возможно полностью имитировать осознанные поступки человека, в то время как сам имитирующий не понимает содержания и смысла этой работы» (стр. 221)[47].

---

[43] **В.Э.:** Это совершенно правильно. Действительно, если я знаю, что такое «сознание», и знаю, что <u>это</u> встроено в компьютер, то мне не нужны никакие проверки «тестом Тьюринга» и все эти сомнения типа «Есть ли сознание? Нет ли сознания?». Тогда я заведомо знаю, что оно есть, и надо проверить лишь – исправно ли оно, как мы проверяем, например, новокупленный холодильник. Мы не сомневаемся в том, что холодильники вообще способны охлаждать продукты; мы проверяем лишь то, исправен ли этот конкретный холодильник. А знания, ставящие всё дело в такой ракурс, дает нам Веданская теория.

[44] **В.Э.:** Профессор Тамберг умер 25 ноября 2008 года.

[45] Penrose Roger, Rouse Ball Professor of Mathematics, University of Oxford. «The Emperor's New Mind. Concerning Computers, Minds, and The Laws of Physics». Oxford University Press, New York, Oxford, 1989.

[46] **В.Э.:** Книга {VITA1}, текст воспроизведен также в {ARTINT}.

[47] **В.Э.:** Это номер страницы в альманахе «Ceļš» («Путь») № 52, где была опубликована статья Ю. Тамберга «Проблема диалога науки и религии в 21-м веке».

.312. Ничего подобного Пенроуз не показывает – он, правда, пытается это показать, но использует для этой демонстрации такие компьютерные программы (своих противников), которые ни в малейшей мере не могут претендовать на «разумную деятельность». Тем самым Пенроуз показал только то, что ТЕ программы не действуют разумно (что специалисту было видно с первого взгляда). Но аргументация Пенроуза ни в малейшей мере не может касаться таких программ (например, сделанных по моему проекту), которые действительно работают разумно. Вообще впечатление такое, что ни Пенроуз, ни его оппоненты – никто – не знает, как нужно строить действительно разумную систему, и поэтому они обсуждают только «всякую чушь», никак не касаясь действительно реального проекта искусственной психики.

.313. Прежде, чем мы продолжим рассмотрение аргументации Пенроуза, ознакомьтесь, пожалуйста, с двумя приложениями к этому моему сочинению: один из них – это мой ответ Пенроузу о тесте Тьюринга (глава из книги, планируемой с названием ROSAE)[48], а второе – оригинальный текст Пенроуза о «китайской комнате»[49].

§27. Реализации и имитации

1999.11.09 15:36 вторник
(раньше на 6 месяцев, 8 дней, 23 часа, 46 минут)

.314. Приложение № 1.

.315. Итак, идея теста Тьюринга была выдвинута в 1950-х годах[50] (во время «первой кибернетической волны»), и в 1960-х, когда этими вещами начал интересоваться я, она звучала со всех сторон; позже большим сторонником теста Тьюринга был наш друг Паулис Кикуст.[51]

.316. С теперешней точки зрения мне эта идея не кажется уже сколь-нибудь ценной и даже представляется, что она свидетельствует об определенном архаизме мышления ее сторонников, о лишь поверхностном знании проблемы, а не о глубоком ее понимании.

.317. Идея теста Тьюринга вообще опирается на кибернетические модели 1950-х годов (Норберт Винер и др.), согласно которым, вот, имеется «черный ящик»; что в нем находится, мы не знаем; у «ящика» есть «вход» и «выход»; на вход подаем одно, на выходе получаем другое...[52] В случае теста Тьюринга ящика два: в одном сидит человек, в другом находится компьютер...

.318. Меня такой уровень рассуждений давно уже не удовлетворяет; «ящик» давно уже для меня не «черный»; я ЗНАЮ, что в нем находится (по крайней мере принципиально знаю), и поэтому лучше говорю о внутренних структурах «ящика», чем о его «входах» и «выходах».

.319. Когда в «ящике» находится человек, то, согласно Веданской теории, там работает определенная (мозговая) операционная система реального времени. Когда же в «ящике» находится компьютер, то сразу возникает вопрос, какого типа программы запущены в этом компьютере. Являются ли они тоже такой же операционной системой реального времени, как и та, что работает в мозге человека, – или это программы совсем другого типа?

---

[48] **В.Э.:** Когда я впервые ознакомился с книгой Пенроуза, я предпринял попытку перевести на латышский язык наиболее важные (с точки зрения Веданской теории) ее главы и написать ответ на них. Это было еще до переписки с Тамбергом. Запланированная тогда книга ROSAE не состоялась, а заготовленные для нее материалы позже вошли в переписку с Тамбергом и в другие сборники (и, вот, дошли и до настоящего издания).

[49] **В.Э.:** В латышской книге в качестве Приложения № 2 {VITA1.370} присутствовала глава Пенроуза «Strong AI and Searle's Chinese room». Здесь это приложение, естественно, опускается, так как здесь и без него имеется полный текст Пенроуза в русском переводе, включая и названную главу.

[50] Теперь, когда существуют поисковые системы Интернета, такие как *Google*, легко выяснить и уточнить, что Тьюринг выдвинул идею теста, названного его именем, в 1950 году в статье «*Computing Machinery and Intelligence*» в связи с дискуссией «Может ли машина мыслить?» для того, чтобы уточнить, что означает мышление машины. Идея теста была взята из игры, в которой один игрок переписывался с двумя другими, один из которых был мужчиной, другой женщиной, и первый игрок должен был угадать, который из его корреспондентов мужчина, а который женщина, причем женщине надо было стараться открыть истину, а мужчине – ее скрыть. Аналогично в оригинальной версии теста Тьюринга человек должен был стараться наблюдателю сказать правду, а машина должна была стараться его обмануть. В более поздних версиях теста это требование отбросили. (*Прим. в издании 2008 года*).

[51] **В.Э.:** Герой «Канторианы» (см. {CANTO = МОИ № 38}, {CANTO2 = МОИ № 39}).

[52] **В.Э.:** Когда я был студентом, нам читали курс лекций под названием «Кибернетика». Этот курс (а также книги, рекомендованные в качестве учебников) начинались с упомянутого «черного ящика».

.320. Чтобы глубже понять этот вопрос и эту разницу, проиллюстрируем проблему на примерах обычных компьютерных операционных систем. Рассмотрим операционную систему WINDOWS или MSDOS, или, скажем, мою DISPOS, которую я сделал и поддерживал с 1976 по 1991 год. Операционную систему нельзя получить, просто складывая вместе отдельные программы. Каждая операционная система имеет определенное «ядро», определенный минимум замкнутых и в идейном отношении завершенных функций, без которого произведение нельзя считать операционной системой. Например, названные операционные системы должны быть способны по крайней мере: 1) поддерживать систему файлов; 2) принять и выполнять команды оператора (человека); 3) запускать под своим управлением различные «пользовательские» программы, предназначенные для этой операционной системы. Если система выполняет это «минимальное ядро», то это операционная система; если не выполняет – то это не операционная система.

.321. Когда этот минимум выполнен, то можно дальше улучшать файловую систему, можно расширять круг выполняемых команд, можно по-разному улучшать работающие под операционной системой программы и их возможности – всё это уже будет дальнейшим развитием операционной системы.

.322. Теперь представим, что какой-то школьник, играя со своим домашним компьютером, написал программу, которая (скажем, при нажатии определенных клавишей) выдает на экране дисплея какие-то сообщения, положим, системы WINDOWS. Тогда наш автор рассказывает, что он уже запрограммировал частичку от операционной системы WINDOWS. На самом деле, это, конечно, НИКАКАЯ НЕ частичка этой операционной системы, а просто чисто внешняя имитация отдельных моментов, – ибо не реализуется основное ядро операционной системы, – необходимый минимум.

.323. Точно так же это и с операционной системой человека: если кто-то сделал компьютерную программу, которая каким-то способом подражает поведению человека в определенные моменты (скажем, поддерживает какой-то диалог), то это еще НИКАКАЯ НЕ частичка от «разума» человека, если при этом не реализуется основное ядро операционной системы человека, – тот минимум, без которого эта система не является и не может являться действительно системой соответствующего типа. Это только чисто внешняя имитация, подобно «*Windows* программе» нашего школьника.

.324. Итак, теперь мы в связи с подражанием другим системам можем различить программы двух типов: 1) действительно реализующие минимальное ядро какой-нибудь операционной системы; и 2) только внешне имитирующие те или иные аспекты какой-нибудь операционной системы. Назовем первые реализациями, а вторые – имитациями.

.325. Теперь посмотрим, ЧТО составляет то минимальное ядро операционной системы человека, без осуществления которого ни одна компьютерная программа не может называться реализацией Системы Человека, а остается только и единственно имитацией.

.326. Это необходимое ядро операционной системы человека составляют следующие «подсистемы»: 1) самопрограммирование – чтобы осуществить любое действие, система должна не просто выполнить уже готовую (данную человеком) программу, а САМА предварительно составить программу этого своего будущего действия, и лишь потом ее выполнить; 2) система должна вести хронику своих предыдущих действий, анализировать эту хронику и результаты своих предыдущих действий, чтобы корректировать дальнейшее самопрограммирование; 3) система должна быть способной усиливать или ослаблять работу отдельных своих аппаратов в зависимости от обстоятельств («раскачивание программ»).

.327. Первая функция будет означать, что система не является «простым автоматом», работающим по заранее заданной программе; вторая функция будет означать, что система имеет «сознание», а третья функция – что она способна на «эмоции».

.328. Реализация этого минимума, этого ядра еще тоже не будет означать, что существует то, что мы в быту называем «разумом». Названный минимум осуществляют все животные – также и рыбы, и ящерицы, и куры; – но если компьютерная программа этого минимума не имеет, то она представляет собой стопроцентную имитацию, и ни о какой ее «разумности» не может быть и речи.

.329. Создание интеллекта, равноценного человеческому интеллекту, следовательно, означало бы: отправляясь с этого минимума, добиться, чтобы система была бы способной путем самопрограммирования создавать для себя (и потом выполнять) программы такого же качества, какие может создавать человеческий мозг.

.330. Теперь мы можем вернуться к тесту Тьюринга. Итак, в одном «черном ящике» находится человек, а во втором – ЧТО? – имитация или реализация?

.331. Если это имитация – и, конечно же, одни лишь имитации до сих пор и обсуждались, ибо никто не знал, КАК создать настоящую реализацию, – если это имитация, то на самом деле уже «*apriori*» ясно, что сколь-нибудь глубокий тест Тьюринга рано или поздно раскроет, что это всего лишь подражание и что «настоящего разума» у системы нет.

.332. Если же это реализация, то насколько высокий уровень нам удалось достигнуть с этой (самопрограммирующейся) системой? Допустим, что нашей системе удалось подняться на уровень самопрограммирования собаки или шимпанзе. Тогда тест Тьюринга моментально раскроет, что эта система в действительности НЕ ЯВЛЯЕТСЯ человеком (так как не умеет даже говорить), в то время, как с хорошей программой имитации тестирующий провозится значительно дольше, прежде чем ее «разоблачит». А в действительности программа имитации отстоит от «настоящего интеллекта» неизмеримо дальше, чем та наша система, которая в самопрограммировании достигла уже уровня шимпанзе: для нее на самом деле остается уже совсем маленький шажок, чтобы она уже оказалась на уровне человека.

.333. Итак, мы видим, что в действительности тест Тьюринга не показывает фактическое положение дел с интеллектом системы; тест был разработан (и далее обсуждался) без сколь-нибудь глубоких знаний о сущности интеллектуальной системы (о содержимом «черного ящика»).

.334. Ясно, что все те программы, о которых в связи с тестом Тьюринга говорит Пенроуз, представляют собой стопроцентные имитации; в книге Пенроуза нет ни малейших следов того, что когда-либо и где-либо кем-либо обсуждались настоящие реализации операционной системы человека; похоже, что никто никогда и нигде не знал, каким образом можно действительно реализовать такую систему, ЧТО для этого необходимо, ЧТО составляет то минимальное ядро системы, без которого имитация безнадежно остается только имитацией.

.335. Поскольку настоящие реализации человеческой операционной системы по-видимому никогда не обсуждались, то и вопрос на самом деле до сих пор стоял так: «Можно ли при помощи имитаций построить человеческий разум?»

.336. Одна часть, оптимисты по природе, но, надо сказать, довольно легкомысленные, отвечали приблизительно так: «Да, конечно, можно! Нагрузим только достаточно много имитаций одну на другую, и всё будет в порядке, – количество перейдет в качество; при достижении определенного уровня сложности спонтанно возникнет разум!»

.337. Вторя часть, и, надо признать, те наиболее глубоко мыслящие – такие как Пенроуз –, скептически качали головами: «Нет, вряд ли можно получить разум, подобный человеческому, если нагромождать одну на другую очень много имитаций; скорее всё-таки, что нельзя...»

.338. В этом споре (составляющем лейтмотив книги Пенроуза и ее глубинный предмет) я, естественно, подтверждаю мнение Пенроуза: «Действительно, операционную систему НЕЛЬЗЯ получить, складывая вместе имитации различных отдельных ее функций». Но разница между Пенроузом и мной состоит в том, что для него этим всё и заканчивается; каким же способом МОЖНО получить операционную систему человека, он не знает, и поэтому начинает думать, что ее нельзя получить вообще, – что и является главным выводом его книги.

.339. Я, напротив, по такому пути уйти не могу, потому что я ЗНАЮ, как надо делать операционные системы вообще и в том числе такие, как у человека. Поэтому весь мой анализ книги Пенроуза сводится – в своей основной сущности – к тому, чтобы изложенной Пенроузом системе взглядов и выводов (которая, может быть, является самой лучшей в условиях, когда НЕ ИЗВЕСТНО, как могла бы быть реализована человеческая операционная система) противопоставить такую систему взглядов и выводов, которая является наиболее естественной в условиях, когда это ИЗВЕСТНО.

.340. При каждой цитате из Пенроуза на всей продолжительности книги[53] первым и решающим шагом у нас всегда будет: сразу посмотреть, как это всё выглядит, если мы всё время держим в уме существование, устройство и работу такой настоящей (а не имитированной!) операционной системы человека.

---

[53] **В.Э.:** Имеется в виду книга ROSAE.

.341. Общий проект такой операционной системы я изложил в письме профессору информатики Фрейбергу,[54] а также в менее систематизированном виде во многих других местах, поэтому здесь всё это будем считать уже известным и не станем повторять.

.342. Когда я указанным способом противопоставляю пенроузовской системе взглядов свою систему, у меня странным образом никогда нет ощущения, что Пенроуз является для меня настоящим противником. Ситуация НЕ такая, что он знал мнение Веданской теории, ознакомился с ним и потом отверг его. Ситуация такова, что он об этом мнении никогда ничего не слышал, не встречался с ним, не оценивал его и поэтому не мог его ни принять, ни отвергнуть. Он как будто блуждает в потемках в поисках того пути, который виден и ясен Веданской теории (этим я никак не хочу уничижить Пенроуза: не все же люди делали операционные системы и не все же должны знать, как их надо делать).[55]

.343. Когда я читаю и перечитываю текст Пенроуза, я не могу освободиться от ощущения, что Пенроуз практически сразу и почти стопроцентно принял бы Веданскую теорию, если только у него была бы возможность с нею ознакомиться. Он никогда не говорит очевидные глупости как доктор Подниекс и те остальные мальчишки из Института математики и информатики Латвийского университета. Мышление Пенроуза тонко и точно, и вся беда заключается только в том, что Веданская теория никогда не лежала перед ним.

.344. Поэтому, хотя излагаемая им система взглядов и предстает противницей Веданской теории и тем самым происходит как будто состязание между Пенроузом и мною («турнир рыцарей ума»[56]), но это состязание – по-настоящему рыцарское, великодушное и даже дружественное.

§28. Витосы и внешний мир

.345. Представим себе, что тот «производитель» (*здесь намек на пример,*[57] *данный Пенроузом – ред.*), который сообщил, что его компьютер (или программа) способен чувствовать, мыслить и т.д., – это я, и что речь идет о <u>моей</u> программе (или, точнее говоря, операционной системе; удобства ради присвоим операционным системам такого типа особое название, скажем, VITOS – от латинского «*vitalis*» – «живой» и английского OS – «*operating system*»; введем также склоняемое по-русски имя нарицательное «витос» для обозначения операционных систем этого типа, т.е. таких, которые выполняют упомянутые в пункте {.326} условия и в которых встроено соответствующее «минимальное ядро»).

.346. Тогда, разбирая аргументы Пенроуза, мы в первую очередь сразу должны задать вопрос: о какой программе здесь идет речь – или о программе имитации, старающейся <u>подражать</u> поведению человека (например, его словам в разговоре), или о программе типа Витоса, которая <u>действительно</u> берет и реализовывает «мышление», «чувствование» и т.д.?

.347. Я могу (в принципе) написать обе эти программы, и принципы их действия (основные алгоритмы) будут совершенно разными.

.348. Если я своей программой стараюсь только подражать человеческому поведению в разговоре (а все слова Пенроуза на самом деле относятся именно к этому варианту), то я в определенной степени являюсь «мошенником»; может быть, без злого умысла: только со спортивным или научным интересом, но всё же «мошенником», старающимся «обмануть» спрашивающего, старающимся своими всё более и более сложными программами имитировать человека. (Как опытный программист могу сказать, что никуда далеко это «мошенничество» уйти не сможет; сколь-нибудь сложный тест Тьюринга имитацию обнаружит).

.349. Если я, напротив, строю <u>настоящий</u> разум и <u>настоящее</u> чувствование, то я должен с самого начала действовать совершенно иначе. Тогда я вкладываю в компьютер только «начальные кирпичики» (в виде некоторых довольно примитивных программных блоков) и – главное – встраиваю в него возможность самопрограммирования. Тогда – возможно в течение многих лет – моя операционная система сама спрограммирует свои высшие уровни; всё время своей жизни она будет накапливать всё новые и новые знания и программы, и потом – скажем,

---

[54] См. {SKATI.472}. (Русский перевод в {POTI-3.472 = МОИ № 43}).

[55] **В.Э.:** Это написано в 1999 году после прочтения Первой книги Пенроуза. Во Второй книге он, конечно, противопоставляет себя взглядам Веданской теории более остро. Всё же в целом сказанное мною в 1999 году остается в силе и теперь (2010).

[56] **В.Э.:** Внешнее оформление книги ROSAE было сделано в виде «турнира рыцарей ума».

[57] **В.Э.:** Это в начале пенроузовского §1.2.

через двадцать лет – ее можно будет пускать на соревнование с человеком в виде теста Тьюринга или в любом другом виде.

.350. Однако, чтобы такой настоящий разум мог бы развиваться, нужно будет, чтобы все эти годы для компьютера была реализована «обратная связь» с внешним миром: чтобы он мог бы «узнать», которые из сгенерированных им самим программ годны, которые негодны и т.д. От того, каким будет этот внешний для него мир, – от этого будет зависеть, что он будет знать, а чего не знать, – и, тем самым, будет ли его легко отличить от оксфордского англичанина.

.351. Попробуйте, например, выполнить тест Тьюринга для того, чтобы отличить современного англичанина начала 21-го века от гвинейского негра времен Васко да Гама в 15-м веке. Ясно, что различить их будет очень легко. Хотя у обоих интеллекты настоящие, но миры, в которых они жили, не имеют почти ничего общего.

.352. Не надо даже искать столь отдаленные примеры: допустим, что мы с Пенроузом спрятались за перегородкой, а в качестве спрашивающей посадили пенроузовскую мадмуазель «*She*» и говорим ей, что один из нас компьютер – пусть она скажет: который? Видимо, она признает Пенроуза человеком, а меня компьютером, потому что я делаю ошибки по-английски, а также не знаю многого об Оксфорде. Но если в качестве спрашивающей посадим Гунту Дроне из Смилтене,[58] то она признает, что человеком являюсь я, а Пенроуз – компьютер. Даже если бы Пенроуза научили бы латышскому языку, то и тогда вскоре обнаружится, что он не знает, что означает «идти по грибы», не может отличить Карлиса Улманиса от Гунтиса Улманиса[59] и думает, что если «кидать по лампе»[60], то лампа разобьётся.

.353. Итак, если мы создаем настоящий интеллект на базе Витоса, то чрезвычайно большое значение будет иметь тот внешний мир, в среде которого мы этот интеллект будем выращивать, обеспечивая всё время ему «обратную связь» назад от этого внешнего мира к выращиваемому интеллекту. Компьютеру, создающему настоящий, неимитированный интеллект, надо давать некоторую «свободу действия» в определенной среде, чтобы мог работать механизм его самопрограммирования.

.354. Эта среда может быть «реальной», т.е. мы могли бы действовать так же, как Природа: посадить наш компьютер в какое-нибудь тело (в механического робота или, может быть, в клонированного человека, настоящий мозг которого по какой-то причине умер, и т.д.). Тогда наш компьютер будет выращивать свой интеллект в условиях, весьма похожих на условия роста людей.

.355. Мы можем также держать компьютер просто на столе, а весь внешний мир для него имитировать какими-то телевизионными сигналами. Тогда он будет действовать в виртуальном мире, похоже на то, как это описал Станислав Лем в своей «Сумме технологии»[61]. Какой мир мы создадим для выращиваемого интеллекта, таким этот интеллект и будет создаваться.

.356. Итак, первая и основная сущность настоящего интеллекта – это самопрограммирование, но самопрограммирование невозможно реализовать без обратной связи, предоставляющей стимулы и критерии для отбора и накопления программ.

.357. Если же мы для этой самопрограммирующейся операционной системы каким-нибудь способом успешно имитировали внешний мир, и этот внешний мир в достаточной степени подражал оксфордскому миру (и, конечно, если «начальные кирпичики» системы были правильными и без дефектов), то в результате будет создан настоящий, неподдельный, индивидуальный интеллект, и можно будет пускать тест Тьюринга (или любой другой тест) сколько угодно: никто этот интеллект не сможет отличить от другого (созданного естественным путем) интеллекта англичанина.

.358. Здесь читатель может возразить так: если уж в этот момент (когда мы пускаем стопроцентно надежный тест Тьюринга) в нашем компьютере существует какая-то программная система (вместе с накопленными за предыдущую жизнь данными), хотя бóльшая часть этой системы (за исключением «начальных кирпичиков») «росла сама» в определенной среде, то ведь мы можем и просто «прямым путем» взять и сделать именно такие же программы без всякого их

---

[58] **В.Э.:** Одна из моих читательниц в 1990-х годах, писавшая мне бурные письма.

[59] **В.Э.:** Карлис Улманис – президент Латвии в 1934–1940 гг.; Гунтис Улманис – его внучатый племянник, президент Латвии в 1993–1999 гг.

[60] **В.Э.:** «Mest pa lampu» – идиоматическое выражение латышского языка, эквивалентное по значению русскому «закладывать за воротник».

[61] Lems Staņislavs. «Summa Technologiae». Sērija «Apvārsnis». Zinātne, Rīga, 1987. (Лем Станислав. *Сумма технологии*. «Мир», М., 1968).

«выращивания», и именно такие же данные «просто записать» в надлежащие места в памяти. Тогда нам не надо будет выращивать «настоящий интеллект» ни в какой среде, а он будет получен «сразу в готовом виде».

.359. Да, конечно, мы можем поступить и так. Но здесь надо помнить о двух вещах. Первая: даже записывая таким способом «напрямую» в компьютер уже готовые программы и готовые данные, мы всё равно «имитируем» определенную среду, определенный «предыдущий опыт» этой системы. Программные системы и накопленные данные оксфордского англичанина и гвинейского негра будут совершенно разными, и разные вещи нам придется записывать в компьютер, в зависимости от того, который из этих двух интеллектов мы хотим туда встроить. Значит, от имитации среды – всё равно, каким способом выполненной, – мы избавиться не можем никак.

.360. Вторая вещь состоит в том, что мы хоть и можем таким путем освободиться от предварительного выращивания программ (до того момента, когда записываем в компьютер готовые и как будто «выращенные» программы), но мы не сможем обойтись без имитации среды в дальнейшей деятельности этого интеллекта. Основной сущностью интеллекта является само-программирование, и это самопрограммирование не сможет работать, если после старта системы мы не дадим ей обратную связь назад от внешнего мира к компьютеру, – если не дадим эту связь соответственно тому, как эта система построена, т.е. тому, как создан ее аппарат само-программирования.

.361. Человеку, который все эти вещи понимает, который действительно может вообразить, представить себе, какими (по крайней мере принципиально) эти программы будут, – такому человеку тест Тьюринга не может казаться важным и существенным. Ясно, что настоящий (и соответствующий планируемой среде тестирования) интеллект всегда пройдет этот тест; ясно также, что ни одна имитация интеллекта его не пройдет (если только задаваемые вопросы будут достаточно глубокими и «коварными»). Идею теста Тьюринга вообще можно выдвигать и (серьезно) обсуждать только те, кто не знают, как работает настоящий интеллект и у кого перед глазами стоят одни лишь имитации. То обстоятельство, что Тьюринг этот тест выдвигает, а Пенроуз его серьезно обсуждает, – одно это обстоятельство уже свидетельствует о том, что никто из них такую действительно интеллектуальную операционную систему себе не представляет, а видят они только имитации и рассуждают только о них.

.362. Речи Пенроуза о том, что придется «тормозить» мощность компьютера, когда вопросы теста Тьюринга будут касаться «сложных арифметических действий»[62] выглядят для специалиста по компьютерам просто наивными. Компьютер может выполнить «сложные арифметические действия» тогда, если запускается соответствующая программа. Программы для арифметических действий могут быть очень разными. Можно, например, выполнить умножение, используя для этого встроенную уже на заводе в микропрограммы компьютера (в систему его инструкций) команду умножения с фиксированной точкой. Для большинства современных компьютеров эта команда работает с числами до 2^{32}. Такое умножение компьютер выполнит несравнимо быстрее человека. Эти арифметические инструкции оперируют битами (числа кодированы в двоичном виде, «естественным» для компьютеров способом).

.363. Для меня такой диапазон чисел был слишком мал, и я сам написал программы, которые выполняют умножение (и другие действия) с числами до 10^{255}. Это умножение работает медленнее, но всё же быстро по сравнению с человеком. Если понадобится, я напишу и программы для арифметических операций с тысячами или миллионами знаков. Программы этой группы оперируют не с битами, а с цифровыми знаками кода ASCII.

.364. Но можно написать и такие программы умножения (и других арифметических действий), которые оперируют не битами, не знаками ASCII, а визуальными образами цифр – так, как это делает человек (например: при помощи сканера вводим в компьютер цифру «4»; компьютер, анализируя изображение, устанавливает, что это «четыре», а не буква «А»; сканируем цифру «2»; компьютер определяет, что это двойка, обращается к таблице умножения,

---

[62] **В.Э.:** В русском переводе: «*..часть задачи программистов состояла бы в том, чтобы в некоторых вещах компьютер казался глупее, чем он есть на самом деле. Если опрашивающая задает сложный арифметический пример, подобный приведенному выше, компьютер должен притвориться, что не в силах на него ответить..*»; в английском оригинале: «*..part of the task for the computer's programmers is to make the computer appear to be `stupider' than it actually is in certain respects. For if the interrogator were to ask the computer a complicated arithmetical question, as we had been considering above, then the computer must now have to pretend NOT to be able to answer it..*».

находит соответствующий знак, изображение «8» и выдает эту картинку в качестве результата). Тогда современный компьютер будет выполнять умножение наверное даже медленнее человека.

.365. Когда в тесте Тьюринга моей операционной системе зададут вопросы арифметической природы, то всё будет зависеть от того, программы какого вида она будет использовать для формирования ответа. Если система выращивалась в определенной среде из «начальных кирпичиков», как это описано выше (или если такое выращивание будет имитировано, записав программы уже готовыми, но такими, какими они могли бы вырасти в определенной среде), то вряд ли системе будут доступны те арифметические программы, которые оперируют битами и знаками ASCII, так как в составе «стартовых кирпичиков» я эти программы системе не дам, раз уж речь идет о выращивании системы, подобной человеку (ибо такие программы не врождены человеку, не кодированы в его ДНК), а сама вырастить такие программы система не сможет; она сможет вырастить только примерно такие же программы умножения, какие работают в мозге человека. Поэтому и в этом аспекте тест Тьюринга не сможет констатировать никаких различий.

.366. Итак, если мы строим настоящий интеллект, а не имитацию, то этот интеллект надо выращивать, позволяя системе самопрограммироваться и в том или ином виде имитируя для нее среду (или предоставляя реальную среду), в которой система будет действовать. Результирующий интеллект будет зависеть от двух факторов: 1) от «начальных кирпичиков», которые мы системе дали; и 2) от той среды, которую мы ей имитировали или дали.

.367. Если двум одинаковым компьютерам мы дадим абсолютно одинаковые «стартовые кирпичики» и будем имитировать для них абсолютно одинаковую среду (скажем, посылая одни и те же видеосигналы), то оба интеллекта будут абсолютно одинаковыми.

.368. Если, напротив, имитируемые среды будут отличаться, то будут отличаться и оба выращенные интеллекта, и каждый из них будет индивидуальным; еще более уникальными интеллекты станут, если будут отличаться также и «начальные кирпичики» (программные блоки, стартовые для самопрограммирования).

.369. В природе, конечно же, всегда отличаются как стартовые условия (комплекты генов), так и – тем более – среда (отличаются события, случающиеся с индивидом, отличаются окружающие его вещи и лица). Поэтому в природе никогда нет двух совершенно одинаковых интеллектов, и каждый человек уникален.

* * *

(Конец вставки; далее продолжается текст Пенроуза)

§1.3. Искусственный интеллект

Очень большой интерес привлекают в последнее время исследования в области, называемой искусственным интеллектом, а часто – сокращенно – «ИИ». Целью этих исследований является научиться максимально возможно имитировать различные аспекты деятельности человеческого разума при помощи машин (как правило, электронных) и, возможно, добиться развития способностей человека в этих направлениях. Есть, по крайней мере, четыре дисциплины, которые проявляют интерес к достижениям в области ИИ. В первую очередь к ним относится робототехника – инженерная отрасль, которая занимается в основном индустриальными механическими устройствами, способными выполнять «интеллектуальные» операции – задачи, разнообразие и сложность которых требует вмешательства и контроля со стороны человека – причем выполнять их со скоростью и надежностью, выходящими за рамки человеческих возможностей, или в неблагоприятных условиях, где жизнь человека будет подвержена опасности. Кроме этого, как с коммерческой точки зрения, так и в целом, представляет интерес развитие экспертных систем, которые позволили бы закодировать самые существенные знания, относящиеся к определенным профессиям – медицинские, юридические и т.п. – в виде пакета компьютерных программ! Возможно ли, чтобы опыт и экспертные оценки специалистов этих профессий были, в самом деле, заменены такими программами? Или единственный результат этих разработок, на который можно надеяться, – это просто длинный список фактической информации с полной системой перекрестных ссылок? Вопрос о том, могут ли компьютеры демонстрировать (или симулировать) полноценную деятельность интеллекта, имеет, несомненно, весьма значительные приложения в социальной сфере. Другой областью, к которой ИИ имеет непосредственное отношение, является психология. Можно надеяться, что попытка смоделировать поведение человеческого мозга (равно как и мозга животного) при помощи электронных устройств – или ее поражение – позволит узнать нечто важное о высшей нервной деятельности. И, наконец, среди оптимистов бытует надежда, что по схожим причинам ИИ мог бы пролить свет на глубокие вопросы философии, дав человеку возможность проникновения в смысл понятия разума.[63]

Как далеко продвинулись исследования ИИ на сегодняшний день? Я едва ли смог бы систематизированно представить здесь все достижения в этой области. В разных уголках мира существует множество активно действующих групп, с работами которых я знаком очень поверхностно. Но справедливости ради необходимо заметить, что, хотя сделано было немало, произвести что-либо, достойное называться подлинным интеллектом, до сих пор никому не удалось. Чтобы дать некоторое представление о предмете обсуждения, я для начала упомяну отдельные ранние (но даже сегодня весьма впечатляющие) достижения, а затем перейду к последним примечательным успехам в области разработки шахматных компьютеров.

Одним из первых устройств ИИ была «черепашка» Грэя В. Уолтера, созданная им в начале 1950-х годов,[64] которая приводилась в движение энергией внутренних батарей и бегала по полу до тех пор, пока они почти полностью не разряжались; после чего она находила ближайшую розетку, подключалась к ней и заряжала их. Когда зарядка заканчивалась, она самостоятельно отсоединялась и продолжала свою прогулку! В дальнейшем было придумано множество подобных механизмов[65] (см., например, Валтц [1982]). Несколько отличное направление развития исследований представляет компьютерная программа Терри Винограда, разработанная в 1972 году, которая могла производить осмысленные действия с набором блоков, разных по форме и цвету, размещая их один над другим в разных сочетаниях и в разном порядке. Поразительно, что, несмотря на эти первые достижения, создание системы контроля[66] даже для простой суставчатой «роботизированной руки», которая должна была в процессе перемещений избегать скопления препятствий, оказалось весьма непростой задачей, хотя стороннему наблюдателю требуемые движения и представлялись совершенно «очевидными».[67] Такой же сложной оказалась и проблема интерпретации зрительно воспринимаемых сцен, которая в общем случае относится к области, где процедуры ИИ даже близко не подошли к реализации того, что мозг человека (и, конечно же, большинства других живых существ) способен делать «без всяких» видимых усилий.[68]

Пожалуй, наиболее созвучна тесту Тьюринга компьютерная программа, разработанная К.М. Колби в середине 1960-х годов, которая имитирует психотерапевта – настолько успешно, что некоторые пациенты обнаружили готовность отдать ей предпочтение перед терапевтом-человеком и рассказать о себе больше, чем при общении с живым врачом. В качестве примера[69] такого общения между компьютерным «психотерапевтом» и пациентом приведем следующий отрывок из диалога (который велся посредством телетайпа):

Терапевт: Чем я могу помочь вам?
Пациент: Я не знаю...
Терапевт: Попробуйте рассказать что-нибудь об этом.
Пациент: Нет.

---

[63] **В.Э.:** Но Пенроуз даже и не подозревает, что «ИИ» имеет колоссальное значение также и для математики, где эти знания полностью переворачивают все представления об основаниях математики.

[64] См. работы Грэгори [1981] и Уолтера [1953].

[65] **В.Э.:** Мне кажется, что все эти «черепашки» скорее помешали развитию полного ИИ, чем способствовали. Именно они и представляют собой самые настоящие имитации разума – а не попытки его реализации. Они направили мысль исследователей и конструкторов к имитациям вместо того, чтобы заняться реализациями. А реализация должна начинаться совсем с другой стороны, с другого конца: с общей принципиальной «блок-схемы» интеллекта – а потом по всем правилам нисходящего проектирования (профессиональный прием разработки больших компьютерных систем) спускаться ко всё более мелким деталям. Но почему-то так никто, насколько мне известно, не поступил, – кроме меня.

[66] **В.Э.:** Плохой перевод; надо: *..системы управления..*

[67] **В.Э.:** Ничего удивительного! Чтобы создать хорошую систему управления рукой, нужно сперва встроить в систему отображение пространства. А это задача действительно сложная – одна из самых сложных во всем ИИ.

[68] **В.Э.** Это потому, что зрительную систему пытались реализовать на обычном офисном компьютере с линейной работой и псевдопараллельными процессами. Но глаза человека и животных же дают нам образец, КАК нужно такие системы строить: большое количество параллельно работающих микропроцессоров, соединенных в сеть (как колбочки и палочки глаза). Поэтому я и говорю, что для ИИ нужны не офисные, а специально сконструированные компьютеры.

[69] Этот пример взят из Дельбрюка [1986].

Терапевт: Можно узнать причину?
Пациент: Я сегодня устал и неважно себя чувствую.
Терапевт: Почему? Есть у вас на этот счет какие-нибудь предположения?
Пациент: Да... Я стар.
Терапевт: Пожалуйста, продолжайте.
Пациент: Мне кажется, что я потерял интерес ко всему происходящему.
Терапевт: Расскажите мне подробнее о ваших ощущениях.
Пациент: У меня есть кое-какие соображения по этому поводу...
Терапевт: У вас есть соображения.
Пациент: Я думаю, что мне недостает семьи.
Терапевт: Что заставляет вас переживать из-за того, что вам недостает семьи?
Пациент: В этом есть какой-то вызов...
Терапевт: Почему это так важно для вас?[70]

Хотя этот диалог может вызвать странное ощущение понимания проблемы пациента компьютером, ничего подобного на самом деле нет: компьютер просто следует довольно простым механическим правилам. (Есть также и «обратная» ситуация, реализованная в системе, где компьютер, имитирующий страдающего шизофренией человека, дает хрестоматийные ответы и симулирует соответствующие симптомы, способные одурачить некоторых студентов-медиков и заставить их поверить в то, что с ними беседует настоящий живой пациент!)

Компьютеры, играющие в шахматы, дают, наверное, наилучший пример поведения, воспринимаемого как «разумное». В действительности, некоторые машины достигли сегодня (к 1989 году) чрезвычайно высокого мастерства игры, если сравнивать с живыми шахматистами, и приближаются к уровню мастера международного класса. (Рейтинг этих машин составляет немногим менее 2300 единиц Эло, тогда как рейтинг чемпиона мира Каспарова, для сравнения, превышает 2700.) В частности, компьютерная программа (для коммерческого микропроцессора *Fidelity Excel*), разработанная Дэном и Кейт Спраклэн, достигла показателя 2110 единиц Эло и была удостоена Шахматной федерацией США звания «Мастера». Еще больше впечатляет программа *Deep Thought*, написанная в основном Хсю (Hsiung Hsu) из университета Карнеги Меллон, рейтинг которой составляет 2500 единиц Эло и которая недавно продемонстрировала замечательное достижение,[71] поделив первое место с гроссмейстером Тони Майлсом на шахматном турнире (Лонгбич, Калифорния, ноябрь 1988 года) и обыграв Бента Ларсена, что можно рассматривать, на самом деле, как первую в истории победу машины над гроссмейстером![72] Сегодня шахматные компьютеры преуспели и в решении шахматных задач, с легкостью превзойдя в этом людей.[73]

Шахматные машины опираются во многом на «книжные знания», помноженные на аккуратность просчета комбинаций. Стоит отметить, что машина в целом «обыгрывает» сравнимого по силе соперника в тех случаях, когда ходы необходимо делать быстро; и «проигрывает» живому противнику, если на каждый ход отпускается достаточное количество времени. Это можно понять, если принять во внимание тот факт, что компьютер принимает решения, опираясь на точные и «быстро разветвляющиеся» вычисления; тогда как преимущество живого шахматиста заключается в его способности производить «суждения», базирующиеся на сравнительно медленной сознательной деятельности по оценке ситуации. Эти человеческие

[70] **В.Э.:** То, что «терапевт»-компьютер здесь не имеет разума, это видно и понятно, но не понятно, почему разума не имеет пациент-человек. Впрочем, разве что потому, что он американец! Меня давно удивляет, до чего глупыми, пустыми и бессмысленными могут быть диалоги (или монологи) в американских фильмах (причем не только игровых, что было бы еще более-менее понятно, но и в документальных!).

[71] В мае 1997 года чемпион мира Г. Каспаров, рейтинг которого превышал 2800 единиц Эло, проиграл компьютерной программе *Deep Blue* со счетом 3,5:2,5. В октябре 2002 года матч между чемпионом мира В. Крамником и программой *Deep Fritz* закончился вничью – 4:4. – *Прим. ред.*

[72] Смотри статьи О'Коннелла [1988] и Кина [1988]. За дальнейшей информацией по компьютерным шахматам я отсылаю читателя к Леви [1984].

[73] Конечно же, сложность большинства шахматных задач рассчитывалась на людей. Возможно, было бы не так уж трудно придумать шахматную задачу, не очень сложную для человеческого существа, но такую, что современные шахматные компьютеры не смогли бы решить и за тысячу лет. (Принцип подобной задачи достаточно очевиден: она должна состоять из очень большого числа ходов. Известны задачи, требующие для решения порядка 200 ходов – более чем достаточно!)

суждения сводятся к тому, чтобы «отбраковать» как можно большее число возможных серьезных вариантов ходов, которые необходимо просчитывать в каждый момент; и при достаточном количестве времени на обдумывание хода такие суждения позволяют производить гораздо более глубокий анализ, чем банальное просчитывание и отбрасывание вариантов, при котором машина не использует подобные суждения. (Такая разница еще более наглядно демонстрируется в сложной восточной игре «го», где число возможностей на каждом ходу значительно больше, чем в шахматах.) Отношение между сознанием и формированием суждений будет центральным моментом в моих дальнейших рассуждениях, особенно в главе 10.

§1.4. Подход к понятиям «удовольствия» и «боли» с позиций ИИ

Согласно одному из распространенных убеждений, ИИ может указать нам путь к своего рода пониманию таких категорий восприятия, как счастье, боль, голод. Возьмем, к примеру, черепашку Грэя Уолтера. Когда ее батареи садятся, ее поведение изменяется и она начинает действовать так, чтобы пополнить запас своей энергии. Здесь есть явная аналогия с тем, как человеческое существо[74] – или любое другое животное – стало бы вести себя, ощутив голод. Похоже, мы не слишком сильно погрешим против языка, если скажем, что черепашка Грэя Уолтера была голодной, когда она действовала упомянутым образом. Некое устройство внутри нее, способное «ощущать» уровень заряда в батареях, заставляло ее переключаться в другой режим функционирования, когда заряд опускался ниже некоторой отметки. Нет причин сомневаться в том, что подобный механизм включается и в голодных животных, но с единственной разницей – изменения модели поведения в этом случае более сложны и деликатны. Вместо простого переключения с одного режима на другой здесь происходит смена направленности действий; и эти изменения усиливаются (до определенной степени) по мере того, как нарастает необходимость восстановить запасы энергии.

Исходя из этого, некоторые приверженцы ИИ утверждают, что такие понятия, как боль или счастье, могут быть смоделированы аналогичным образом. Давайте упростим задачу и будем рассматривать линейную шкалу «чувств», простирающуюся от крайней «боли» (отметка: –100) до абсолютного «удовольствия» (отметка: +100). Представим далее, что у нас есть устройство – какая-нибудь машина, предположительно электронная, – которая располагает средствами для регистрации собственного (условного) показателя «боль–удовольствие», который я буду называть «бу-показатель». Устройство это должно иметь определенные модели поведения и входные данные, как внутренние (типа состояния батарей), так и внешние. Идея заключается в том, что все действия машины должны быть подчинены критерию максимизации ее бу-показателя. Факторов, влияющих на его величину, может быть множество. Мы, конечно же, можем сделать одним из них уровень заряда батарей, так, чтобы низкий уровень давал отрицательный вклад, а высокий – положительный; но могут существовать и другие факторы. Возможно, наше устройство несет на себе солнечные батареи, которые дают альтернативный источник энергии, при активации которого аккумуляторы перестают использоваться. Мы можем задать такую программу действий, при которой движение к свету будет немного увеличивать бу-показатель устройства – что оно и будет стремиться делать при отсутствии иных факторов. (Хотя, на самом деле, черепашка Грэя Уолтера, как правило, избегала света!) Ему потребуются какие-нибудь средства для выполнения вычислений, позволяющих оценивать последствия тех или иных действий в терминах величины бу-показателя. В дополнении к этому оно может уметь вводить вероятностные веса, так, чтобы в зависимости от достоверности исходных данных вычисления давали больший или меньший вклад в бу-показатель.

Помимо этого нашему устройству необходимо будет задать еще и дополнительные «цели», отличные от поддержания уровня его энергетических запасов, поскольку в противном случае мы не сможем отделить «боль» от «голода». Естественно, было бы слишком требовать от нашего механизма способности к размножению, поэтому давайте пока забудем о сексе! Но, возможно, мы могли бы имплантировать ему «желание» общения с аналогичными устройствами, приписывая таким встречам положительное значение бу-показателя. Или же мы можем заложить в него чистую «жажду знаний», когда даже простое накопление фактов об окружающем мире

[74] **В.Э.:** Ну что за перевод – «человеческое существо»?! Это же просто «человек». Это английский язык такой бедный, что в нем нет слова «человек», и они вынуждены говорить либо «*man or woman*», либо «*human being*». А русский язык же велик и могуч – зачем подражать тем бедняжкам?!

имело бы положительный эффект на величину бу-показателя. (Действуя из эгоистических побуждений, мы могли бы сделать так, что этот показатель увеличивался бы в результате оказания нам различных услуг – в точности, как при создании робота-слуги!) Можно было бы расценивать такой подход к назначению «целей» как искусственный, поскольку мы руководствуемся здесь разве что своими капризам. Но, в действительности, это не слишком уж отличается от способа, которым нам как индивидуумам определяются «цели» в процессе естественного отбора, где главенствующим фактором является необходимость распространять наши гены.

Предположим теперь, что мы благополучно создали наше устройство, учтя все вышеизложенные требования. Но есть ли у нас основания утверждать, что оно будет и вправду чувствовать удовольствие при положительном, а боль – при отрицательном значениях бу-показателя? С позиций ИИ (т.е. с операционалистской точки зрения), мы должны судить об этом просто по тому, как устройство себя ведет. Раз оно действует с таким расчетом, чтобы увеличить свой бу-показатель настолько, насколько это возможно (и удерживать его на этом уровне максимально продолжительное время), и, соответственно избегать его отрицательных значений, то было бы разумным <u>определить</u> чувство удовольствия как степень положительности бу-показателя, а чувство боли – как степень его отрицательности. «Обоснованность» этого метода определения вытекает из полного сходства такого поведения с реакциями человека на удовольствие или боль. Конечно же, человеческие существа, как известно, далеко не так примитивны: иногда мы, кажется, намеренно не избавляемся от боли или избегаем некоторых удовольствий. Очевидно, что в наших действиях мы руководствуемся гораздо более сложными критериями (см. Деннетт [1978]). Но в качестве очень грубой аппроксимации можно считать, что все-таки в большинстве случаев мы стараемся избегать боли и получать удовольствие. Для операционалиста этого было бы достаточно, чтобы оправдать – в таком же приближении – идентификацию бу-показателя нашего устройства с его рейтингом по шкале «боль–удовольствие». Возможность установления подобных соответствий – одно из направлений теории ИИ.

Вопрос, который мы должны задать: правда ли, что наше устройство может по-настоящему <u>чувствовать</u> боль, если его бу-показатель отрицателен, и удовольствие в противном случае? Да и способно ли оно чувствовать хоть что-нибудь вообще? Операционалист, конечно, сказал бы «Естественно, да!»; либо отбросил бы этот вопрос как бессмысленный. Но мне представляется, что здесь <u>есть</u> серьезный и сложный вопрос, который необходимо рассмотреть.[75] На наши действия влияет множество разнообразных факторов. Некоторые из них осознанные, как боль или удовольствие, тогда как другие мы не воспринимаем сознанием. Это наглядно иллюстрируется примером человека, касающегося раскаленной плиты. Приводится в действие механизм, который заставляет человека непроизвольно отдернуть руку еще до того, как он почувствовал боль. Вполне может оказаться, что такие спонтанные действия гораздо ближе по своей природе к реакциям нашего устройства, обусловленным его бу-показателем, чем те, которые действительно вызваны болью или удовольствием.

При описании поведения машин часто – и, обычно, в шутку – используются «человеческие» понятия: «Моя машина не хотела заводиться сегодня утром»; или «Мои часы до сих пор думают, что они идут по калифорнийскому времени»; или «Мой компьютер заявляет, что не понимает последнюю команду и не знает, что делать дальше». Конечно же, мы никоим образом не подразумеваем, что машина <u>действительно</u> может чего-либо <u>хотеть</u>, часы – что-то <u>думать</u>, а

---

[75] **В.Э.:** Из всего, о чем Пенроуз здесь говорит, действительную проблему я вижу только с чувством боли и отчасти голода и сексуального влечения («чисто физического»). Все психические состояния, такие как страх, счастье, тоска, любовь и т.д. легко объясняются состояниями (мозговой) программной системы, и проблем здесь нет. А с названными тремя чувствами (которые, как легко догадаться, представляют собой филогенетически самые древние механизмы) действительно какая-то проблема чувствуется, и дело обстоит так же, как с ощущением «красного», которое я разобрал в комментарии к книге {PENRS1 = МОИ № 17}, стр.44). То есть, в конце концов нам приходится предполагать, что всё то объективное, что мы можем под этими чувствами найти, совпадает с самим чувством. Возможно, проблему порождает как раз то обстоятельство, что эти чувства являются филогенетически самыми древними, их первоначальная отработка происходит в другой части мозга (или вообще нервной системы), и в операционную систему нашего «сознания» они приходят как сигналы <u>внешние</u>. Это, пожалуй, основной момент здесь: всё, что для операционной системы нашего сознания является внутренним, объясняется легко, – а всё, что для нее приходит как сигналы внешние, выглядит труднопонимаемыми «ощущениями». (А если это так, то ведь и понятны условия, при которых компьютер начнет «ощущать»!).

компьютер[76] – о чем бы то ни было заявлять, а также понимать или даже знать, что он делает. Тем не менее подобные выражения могут быть поистине информативными и способствовать нашему пониманию, при условии, что мы их будем рассматривать только в том духе, в котором будем их произносить, а не в буквальном смысле слова. Я всегда занимаю в целом аналогичную позицию по отношению к различным заявлениям сторонников ИИ о том, что сконструированные человеком устройства могут обладать характеристиками сознания – безотносительно от того, что под этим подразумевается! Если я согласен говорить, что черепашка Грэя Уолтера может быть голодной, то только лишь в полушутливом тоне. И если я готов использовать такие термины типа «боль» или «удовольствие», связывая их с бу-показателем некоторого устройства, как я это делал выше, то единственная причина этому заключается в том, что эти выражения облегчают мое понимание поведения устройства благодаря определенным аналогиям с моим собственным поведением и состояниями сознания. Причем здесь я ни в коем случае не подразумеваю, что эти аналогии особенно близки, или что не существует прочих – нерегистрируемых сознанием – явлений, которые влияют на мое поведение гораздо более схожим образом.

Я надеюсь, что читателю мое мнение достаточно ясно: я считаю, что проблема понимания свойств сознания гораздо более многогранна, чем можно извлечь непосредственно из экспериментов с ИИ. Тем не менее, я уверен в необходимости признания этой области исследований и уважительного отношения к ней. При этом я не собираюсь утверждать, будто бы достижения в задаче моделирования действительного интеллекта велики (если они вообще есть). Но нужно всегда помнить о том, что сам предмет очень «молод».

Компьютеры станут быстрее, будут обладать высокоскоростным доступом к более вместительным устройствам хранения информации, большее количество логических элементов, и научатся выполнять большее число операций параллельно. Улучшится логическая структура и техника программирования. Эти машины – носители философии ИИ – значительно и всесторонне улучшат свои возможности. Более того: сама философия отнюдь не является абсурдной по самой своей сути. Возможно, что человеческий разум может и в самом деле быть смоделирован[77] с очень большой степенью точности при помощи электронных компьютеров – тех самых, которыми мы располагаем сегодня и принципы действия которых нам уже понятны, – но более мощных по своим характеристикам, чье появление в ближайшие годы вполне предсказуемо. Вероятно даже, что эти устройства и вправду будут разумными; возможно, они будут думать, чувствовать и иметь собственный интеллект. Или же, наоборот, они не будут разумными, и потребуются какие-то новые принципы, в которых мы сегодня остро нуждаемся. В этом-то и заключается вопрос, от которого нельзя просто отмахнуться. Я постараюсь предоставить в ваше распоряжение факты так, как я их вижу; затем я приведу свои собственные соображения на этот счет.

§1.5. Сильный ИИ и китайская комната Серла

Существует точка зрения, называемая сильный ИИ,[78] которая занимает весьма радикальную позицию по этим вопросам.[79] Согласно теории сильного ИИ, не только вышеупомянутые устройства будут разумны и наделены интеллектом – свойства разума могут быть присущи логическим действиям любого вычислительного устройства, даже простейших из них, механических, одним из которых является, например, термостат.[80] Основная идея заключается в том, что умственная деятельность – это просто выполнение некоторой хорошо определенной последовательности операций, часто называемой алгоритмом. Далее я уточню это понятие. А пока нам будет достаточно определить алгоритм как своего рода вычислительную процедуру. В случае термостата алгоритм чрезвычайно прост: устройство фиксирует повышение или понижение температуры по отношению к заданной величине и размыкает или замыкает цепь, соответственно. Алгоритм, соответствующий более-менее нетривиальной деятельности голов-

[76] По состоянию дел на 1989 год!

[77] **В.Э.:** Не надо моделировать! Надо выполнить работу человеческого мозга на другом устройстве.

[78] **В.Э.:** В оригинале «Strong AI»; но мне кажется, что правильнее было бы переводить это как «строгий ИИ»: подразумевая, что это «ИИ по строгому определению».

[79] Везде в этой книге я использую термин Серла «сильный ИИ» для обозначения этой радикальной точки зрения просто чтобы быть точным. Слово «функционализм» часто применяется по отношению к такому же по сути воззрению, но, наверное, не всегда корректно. Этой точки зрения придерживаются Мински [1968], Фодор [1983], Хофштадтер [1979], Моравец [1989].

[80] См. работу Серла [1987] в качестве примера такого утверждения.

ного мозга, должен быть гораздо более сложноструктурированным, но – согласно концепции сильного ИИ – это будет всё же алгоритм. Он будет очень значительно отличаться от простейшего алгоритма термостата по степени сложности, но не обязательно будет иметь принципиальные отличия. Таким образом, с точки зрения сильного ИИ, существенная разница между деятельностью человеческого мозга (включая все проявления сознания) и работой термостата состоит единственно в этой самой усложненности (или, возможно, «структуре более высокого порядка», или «способности обращения к самому себе», или в любом другом свойстве, которое можно приписать алгоритму), имеющей место в первом случае. И, что более важно, все свойства ума – мышление, способность чувствовать, интеллект, понимание, сознание – должны рассматриваться, согласно этому подходу, просто как разные аспекты сложной деятельности; иными словами, они есть не более, чем свойства алгоритма, выполняемого мозгом.[81] Достоинства любого конкретного алгоритма заключаются в его «технических характеристиках», таких как точность результатов, область применимости, экономичность и скорость выполнения. Алгоритм, нацеленный на подражание тому, что, как предполагается, действует в мозге человека, должен быть невообразимо сложным.[82] Но если такой алгоритм для мозга существует – а это как раз то, что с уверенностью утверждают поборники идеи сильного ИИ, – то он в принципе мог бы быть запущен на компьютере. В сущности, он мог бы выполняться на любом современном компьютере общего назначения, если бы не имеющиеся ограничения по скорости и пространству для хранения данных. (Обоснование этого замечания будет дано позднее, когда мы перейдем к рассмотрению универсальной машины Тьюринга.) Предполагается, что такие ограничения будут сняты с появлением в недалеком будущем мощных быстродействующих машин. Тогда такой алгоритм, если он будет открыт, мог бы, вероятно, пройти тест Тьюринга. И как только он будет запущен, считают сторонники сильного ИИ, он будет сам по себе испытывать чувства, обладать сознанием, быть разумом.[83]

Далеко не каждый согласится с тем, что разумные состояния и алгоритмы можно считать идентичными в указанном контексте. Наиболее остро критиковал эту точку зрения американский философ Джон Серл [1980, 1987]. Он приводил в пример ситуации, когда должным образом запрограммированный компьютер проходил упрощенную версию теста Тьюринга, и всё же – он подкрепляет эти выводы очень сильными аргументами – «понимание» как свойство интеллекта полностью отсутствовало. Один из таких примеров базируется на компьютерной программе, разработанной Роджером Шенком (Шенк, Абельсон [1977]). Задачей программы была имитация понимания простых историй типа: «Мужчина вошел в ресторан и заказал гамбургер. Когда гамбургер принесли, оказалось, что он сильно подгорел, и рассерженный мужчина выскочил из ресторана, не заплатив по счету и не оставив чаевых». В качестве второго примера можно взять другую историю: «Мужчина вошел в ресторан и заказал гамбургер. Когда его принесли, мужчина остался им очень доволен. И, покидая ресторан, он дал официанту щедрые чаевые перед тем, как

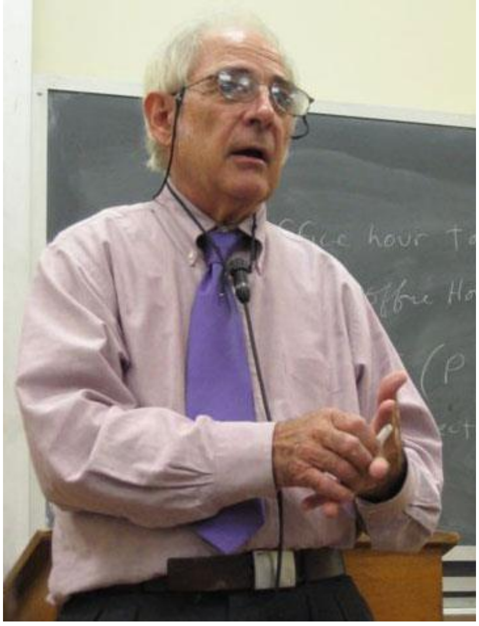
---

[81] **В.Э.:** «Чисто теоретически», конечно, можно рассматривать всё, происходящее в мозге, как разворачивание одного грандиозного, «чрезвычайно сложного» алгоритма, так как это действительно причинно-следственные цепочки нашего строго детерминированного мира. Но такой подход лишает нас возможности что-либо понять в этом алгоритме-гиганте. Чтобы понимать мир, мы должны его структурировать: выделять в нем объекты и находить связи между этими объектами. Чем удачнее мы это сделаем, тем глубже мы поймем изучаемый мир. Поэтому вместо одного «гигантского алгоритма» мы должны рассматривать множество отдельных алгоритмов, по которым работают отдельные мозговые программы. Тогда картина выглядит так, что не задан наперед на всю жизнь один определенный «алгоритм мозга», а (в процессе самопрограммирования) одни программы порождают другие программы (с их алгоритмами), шлифуют, изменяют, оттачивают их алгоритмы. Алгоритмы (вместе с программами, в которых они воплощены) в мозге непрерывно появляются и исчезают, а во время своего существования – изменяются. Такая картина (модель) более продуктивна (дает возможность изучать мозговые программы и их взаимодействие) и более соответствует действительному положению вещей, нежели представление об одном гигантском «алгоритме мозга».

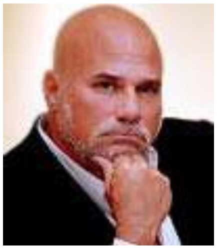
[82] **В.Э.:** Это если мы рассматриваем всю деятельность мозга на протяжении всей жизни как выполнение одного гигантского алгоритма. Но, как я уже сказал, такая модель не способствует пониманию вещей. Если же мы принимаем модель, согласно которой в мозге действует огромное количество различных алгоритмов (миллиарды и миллиарды), которые появляются (в результате самопрограммирования) и исчезают из мозга (если признаны негодными), то каждый отдельный из таких алгоритмов в общем-то довольно прост.

[83] **В.Э.:** Система будет «*испытывать чувства, обладать сознанием, быть разумом*» тогда, если она выполнит минимальное ядро Витоса.

заплатить по счету». Чтобы проверить «понимание» этих историй компьютером, его «попросили» определить, съел ли мужчина гамбургер в каждом отдельном случае (факт, который не был упомянут в тексте явным образом). На этот простой вопрос к таким простым историям компьютер может дать ответ, совершенно неотличимый от того, что дал бы англоговорящий человек, а именно: «нет» в первом случае и «да» – во втором. Так что в этом, очень узком, смысле машина уже прошла тест Тьюринга!



John Rogers Searle[84]



Roger Schank, born 1946

Вопрос, к которому мы должны далее обратиться, будет таким: действительно ли подобный положительный результат указывает на истинное понимание, демонстрируемое компьютером – или, возможно, заложенной в него программы? Как аргумент в пользу отрицательного ответа на этот вопрос, Серл предлагает свою концепцию «китайской комнаты». Он сразу же оговаривает, что истории должны рассказываться на китайском, а не на английском языке – совершенно несущественная замена – и что все команды для компьютерного алгоритма в этом конкретном случае должны быть представлены набором (английских) инструкций для работы со счетами, на которые нанесены китайские символы. Проводя мысленный эксперимент, Серл представлял, что он сам выполняет все манипуляции внутри запертой комнаты. Последовательность символов, описывающая истории, и вопросы к ним подаются в комнату через небольшие прорези. Никакой другой информации извне не допускается. В конце, когда все действия выполнены, последовательность, содержащая ответ, выдается из той же прорези наружу. Поскольку все эти операции есть не что иное, как составляющие процедуры выполнения алгоритма по программе Шенка, то эта последовательность должна содержать просто китайские символы, означающие «да» или «нет» и дающие корректный ответ на вопрос, который – как, собственно, и сама история – был изложен по-китайски. При этом Серл недвусмысленно дает понять, что он не знает ни слова по-китайски, и посему не имеет ни малейшего представления о содержании рассказанных историй. Тем не менее, выполнив ряд действий, составляющих алгоритм Шенка (инструкции к которому были даны ему на английском языке), он справился бы с задачей не хуже китайца, способного без труда понять эти истории. Довод Серла – и весьма сильный, по моему мнению, – заключается в том, что простое выполнение подходящего алгоритма еще не говорит о понимании. (Воображаемый) Серл, запертый в китайской комнате, не понимает ни на йоту, о чем идет речь в этих историях!

Против доказательства Серла был выдвинут ряд возражений. Я изложу здесь только те из них, которые – на мой взгляд – имеют серьезное значение. Прежде всего, фраза «не знает ни слова», если рассматривать ее в вышеприведенном контексте, является не вполне корректной. Понимание относится не только к отдельным словам, но и к определенным шаблонам. И при выполнении подобных алгоритмов можно в достаточной степени разобраться в структурах, которые составлены из символов, значение каждого из которых в отдельности останется непонятным. Например, китайский иероглиф, соответствующий «гамбургеру» (если он вообще существует), можно заменить на название какого-нибудь другого блюда, допустим, «чоу мейн»[85], существенно не изменив при этом содержание истории. Однако, мне все-таки кажется, что настоящий смысл историй (даже если считать такие подстановки незначительными) едва ли «дойдет» до того, кто будет просто скрупулезно выполнять шаг за шагом подобные алгоритмы.

---

[84] **John Rogers Searle** (born July 31, 1932 in Denver, Colorado) is an American philosopher and the Slusser Professor of Philosophy and Mills Professor of Philosophy of Mind and Language at the University of California, Berkeley (UC Berkeley).

[85] Чоу мейн (англ. *chow mein*) – распространенное китайское блюдо на основе жареной лапши. – *Прим. ред*.

Во-вторых, нужно всегда помнить о том, что выполнение даже сравнительно простой компьютерной программы оказывается в большинстве случаев длительным и трудным процессом, если за него берется человек, манипулирующий символами. (В конце концов, именно по этой причине мы доверяем такие действия компьютерам!) Если бы Серл в самом деле выполнял указанным выше способом алгоритм Шенка, то ему для ответа на совсем простой вопрос понадобились бы дни, месяцы, а то и годы изнурительно однообразной работы – не слишком правдоподобное занятие для философа! Однако, это не представляется мне таким уж серьезным возражением, поскольку здесь мы рассматриваем вопрос в принципе и не касаемся технических деталей. Больше затруднений вызывает предположение о наличии компьютерной программы, способной сравниться с человеческим мозгом и, тем самым, безупречно пройти тест Тьюринга. Любая подобная программа должна быть невероятно сложной. Нетрудно вообразить, что действие такой программы, необходимое для нахождения ответа даже на сравнительно простой вопрос теста Тьюринга, состояло бы из столь большого количества шагов, что ни для одного человеческого существа выполнение соответствующего алгоритма за период, равный средней продолжительности жизни, было бы невозможным.[86] Так ли это на самом деле – трудно сказать, не имея подобной программы в своем распоряжении.[87] Но, в любом случае, вопрос о чрезвычайной сложности (программы), по-моему, игнорировать нельзя. Понятно, что мы говорим о принципиальной стороне дела; и всё же мне не кажется таким уж невероятным существование некоторой «критической» степени сложности алгоритма, которой необходимо достигнуть, чтобы алгоритм начал обладать качествами разума. Возможно, это критическое значение так велико, что ни один алгоритм, имеющий столь сложную структуру, не может быть выполнен вручную ни одним человеческим существом, как то предлагает Серл.

Сам Серл в качестве контраргумента к последнему возражению предлагает заменить фигурирующего ранее «жильца» (самого себя) китайской комнаты – целой командой не понимающих китайский язык манипуляторов символами. Чтобы сделать это число достаточно большим, он даже допускает возможность замены своей комнаты всей Индией, где всё население (кроме понимающих китайский!) будет производить действия над символами. Хотя с практической точки зрения это было бы безумием, принципиально это далеко не абсурдная модель, которая не вносит существенных изменений в первоначальные выводы: те, кто манипулирует символами, по-прежнему не понимают содержание историй, вопреки утверждениям сторонников сильного ИИ о том, что простое выполнение подходящего алгоритма вызвало бы возникновение присущего интеллекту свойства «понимания». Однако, теперь это возражение оттесняется на задний план другим, кажущимся серьезнее: что, если эти индийцы более похожи на отдельные нейроны в человеческом мозгу, чем на этот мозг в целом? Никто никогда не будет ожидать от нейронов, чье возбуждение, по-видимому, является центральным механизмом умственной деятельности, чтобы они сами понимали, о чем думает их «хозяин» – так почему же индийцы должны понимать китайские истории? Серл парирует это возражение, указывая на явную абсурдность представления об Индии как реальной стране, понимающей некую историю, в то время как всё ее население не имеет о ней ни малейшего понятия. Страна, говорит он, как и термостат или автомобиль, не «занимается» пониманием – это прерогатива индивидуумов, проживающих на ее территории.

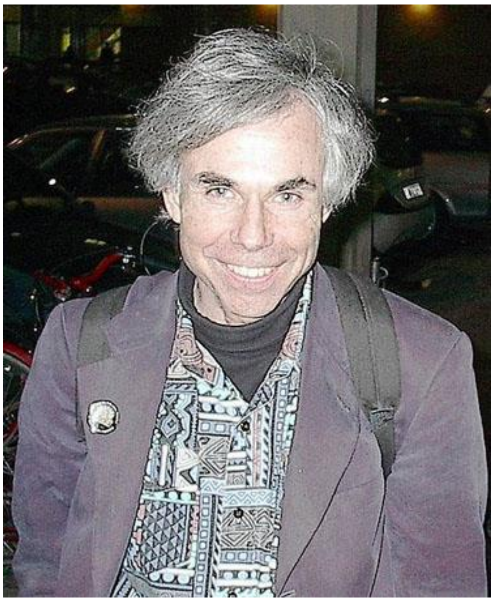
Этот аргумент выглядит значительно слабее предыдущего. Я думаю, что доказательство Серла наиболее убедительно в случае одного исполнителя алгоритма, где мы должны ограничиться алгоритмом, чья степень сложности допускает его выполнение за время, не превышающее нормальную продолжительность человеческой жизни. Я не рассматриваю этот аргумент как непреложное свидетельство того, что не существует никакого бестелесного «понимания», ассоциируемого с процессом выполнения алгоритма людьми, чье присутствие никак не влияет на их собственное сознание. Однако, я бы скорее согласился с Серлом, что эта

---

[86] **В.Э.:** Путанное предложение переводчика; надо либо «*ни для одного .. не было бы возможным*», либо «*для любого .. было бы невозможным*».

[87] Дуглас Хофштадтер в своей критике оригинальной работы Серла (так, как она перепечатана в *The Mind's I*) возражает, что ни одно человеческое существо не в состоянии «разобраться» в полном описании разума другого человека из-за большой сложности. И это действительно так! Но мне кажется, что идея не в этом. Ведь выполнить нужно будет только ту часть алгоритма, которая должна отвечать какому-то одному мыслительному процессу. Таким могло бы оказаться некое мгновенное «осознание» при ответе на вопрос теста Тьюринга, или даже что-нибудь еще более простое. А кто сказал, что подобное действие с необходимостью потребовало бы выполнения алгоритма невообразимой сложности?

возможность представляется, мягко говоря, малоправдоподобной. Мне сдается, что довод Серла весьма убедителен, хотя и не является решающим. Он с очевидностью демонстрирует, что алгоритм такой степени сложности, которой обладает компьютерная программа Шенка, не может иметь какого бы то ни было понимания выполняемых задач; также из него предположительно следует (и не более того), что ни один алгоритм, независимо от сложности его структуры, не может сам по себе воплощать настоящее понимание – вопреки утверждениям поборников сильного ИИ.



Douglas Richard Hofstadter[88]

Существуют, на мой взгляд, и иные очень серьезные проблемы, связанные с сильным ИИ. Согласно этой точке зрения, единственное, что имеет значение – это алгоритм. И совершенно неважно, кто приводит его в действие: человеческий мозг, электронный компьютер, целое государство индийцев, механическое устройство из колесиков и шестеренок или система водопроводных труб. В рамках этой теории существенным для воплощения заданного «состояния разума» является сама логическая структура алгоритма, а его физическая реализация никакой роли не играет. Но, как указывает Серл, это может привести к определенной форме дуализма. Дуализм – это философское мировоззрение, апологетом которого был в высшей степени влиятельный философ и математик XVII века Рене Декарт, утверждавший, что существуют две различные субстанции: «разумная субстанция» и обычная материя. Влияют ли они друг на друга, и если да, то каким образом – это уже отдельный вопрос. Ключевое положение этой точки зрения заключается в гипотезе о том, что «разумная субстанция» не может состоять из материи обычной и способна существовать независимо от нее. «Разумная субстанция» в представлениях сильного ИИ – это логическая структура алгоритма. Как я отмечал выше, ее физическое воплощение не имеет никакого значения. Алгоритм обладает неким бесплотным существованием, никак не связанным с конкретной физической реализацией. Насколько серьезно мы должны воспринимать такой вид существования – вопрос, к которому мне придется вернуться в следующей главе. Он представляет собой часть более глобального вопроса о платонистической реальности абстрактных математических объектов. Пока же я обойду эту общую тему стороной и отмечу только, что сторонники сильного ИИ, по-видимому, принимают всерьез возможность подобного существования в случае алгоритмов, полагая, что те являются самой «сущностью» их мыслей, чувств, понимания и сознательного восприятия. В связи с этим Серл указал на примечательный в своей ироничности факт: теория сильного ИИ может привести к крайней форме дуализма – к той точке зрения, к которой сторонники сильного ИИ менее всего хотели бы иметь отношение!

Эта дилемма просматривается в рассуждениях, предложенных Дугласом Хофштадтером [1981] – убежденным сторонником сильного ИИ – в диалоге с названием *Беседа с мозгом Эйнштейна*. Хофштадтер выставляет на обозрение книгу, имеющую абсурдно большие размеры и содержащую, по его утверждению, полное описание мозга Альберта Эйнштейна. Идея такова: на любой вопрос, который кто-либо пожелал бы задать Эйнштейну, можно получить ответ в точности такой, каким был бы ответ живого Эйнштейна, если просто листать книгу и тщательно следовать всем приведенным в ней инструкциям. Конечно же, слово «просто» здесь совершенно неуместно, как то особо оговаривает сам Хофштадтер. Ведь смысл его утверждения иной: принципиально эта книга полностью эквивалентна (в операционалистском смысле теста Тьюринга) до смешного медленной «версии» настоящего Эйнштейна. Тем самым, если следовать положениям теории сильного ИИ, эта книга должна была бы думать, чувствовать, понимать и осознавать в точности так, как это делал бы сам Эйнштейн, только невероятно медленно (так что

---

[88] **Douglas Richard Hofstadter** (born February 15, 1945 in New York, New York) is an American academic whose research focuses on consciousness, thinking and creativity. He is best known for *Gödel, Escher, Bach: an Eternal Golden Braid*, first published in 1979, for which he was awarded the 1980 Pulitzer Prize for general non-fiction. Hofstadter is the son of Nobel Prize-winning physicist Robert Hofstadter. He grew up on the campus of Stanford University, where his father was a professor.

для этого «книго-Эйнштейна» внешний мир казался бы мелькающим перед ним с огромной скоростью). И естественно, что книга, представляющая из себя частную реализацию алгоритмизованной «сущности» Эйнштейна, была бы как раз-таки самим Эйнштейном.

Но тут возникает другая трудность. Книгу могут не открыть ни разу – или же, напротив, над ней будут корпеть многочисленные студенты и искатели истины. Как книга «поймет» разницу между этими двумя крайностями? Возможно, книгу даже не понадобится открывать, если в ход будет пущено считывание информации при помощи рентгеновской томографии или какое-нибудь другое технологическое чудо-средство. Осознает ли Эйнштейн, что книга изучается подобным образом? Будет ли он знать о двух попытках найти с его помощью ответ на один и тот же вопрос, если он был задан дважды, разными людьми и в разное время? Или это вызовет две разделенные по времени копии одного и того же состояния осознания? Возможно, акт осознавания будет иметь место только в случае изменений, произошедших с книгой? В конце концов, мы обычно осознаем нечто, когда получаем о нем информацию извне, которая воздействует на наши воспоминания и, естественно, несколько изменяет состояние нашего ума. Если это так, то означает ли это, что именно (соответствующие) изменения алгоритмов (здесь я рассматриваю хранилище информации как часть алгоритма) должны приниматься за события, происходящие в процессе умственной деятельности – а не само выполнение (хотя, быть может, и оно тоже) алгоритмов? Или же «книго-Эйнштейн» способен полностью осознавать себя даже в том случае, когда его никто не будет изучать и ничто не потревожит? Хофштадтер затрагивает некоторые из этих вопросов, но на большинство из них он даже не пытается по-настоящему ответить или хотя бы подробно разобраться с ними.

Что значит «запустить алгоритм» или «реализовать его физически»? Будет ли изменение алгоритма как-нибудь отличаться от его замены на другой алгоритм? И как же всё это, черт побери, связано с нашими чувствами и осознаванием?! Читатель (если только он не принадлежит к лагерю сторонников сильного ИИ) может удивиться, видя сколько времени я уделяю такой заведомо абсурдной идее. Но я-то, на самом деле, не считаю ее изначально абсурдной – только лишь неверной! Некоторые рассуждения, на которые опирается теория сильного ИИ, я считаю достаточно убедительными и попытаюсь обосновать свое мнение ниже. В некоторых идеях – если их модифицировать подходящим образом – есть, на мой взгляд, определенная привлекательность, которую я также постараюсь передать. Более того: как мне кажется, те самые контраргументы, которые приводит Серл, в свою очередь тоже содержат ряд серьезных головоломок и кажущихся нелепостей – хотя, в какой-то степени, я с ним и согласен!

Серл в ходе своих рассуждений неявным образом признает, что сегодняшние электронные компьютеры, снабженные значительно увеличенными быстродействием и размерами устройств хранения информации с высокой скоростью обмена данными (и, возможно, параллельным выполнением операций), вполне могли бы в обозримом будущем успешно пройти тест Тьюринга. Он готов признать утверждение сторонников сильного ИИ (и многих других «научных» точек зрения), что мы «*просто конкретные экземпляры реализации некоторого числа компьютерных программ*». Более того, он соглашается и с тем, что: «*Конечно, наш мозг является цифровым компьютером. Поскольку всё есть цифровые компьютеры, то и мозг – тоже*»[89]. Серл полагает, что разница между действием человеческого мозга (который может иметь разум) и электронным компьютером (который, как он утверждает, такого свойства не имеет), когда они выполняют один и тот же алгоритм, состоит исключительно в материальной конструкции того и другого. Он заявляет – правда, не давая этому никакого обоснования – что биологические объекты (мозг) могут обладать «ментальностью» и «семантикой», которые он считает основополагающими для умственной деятельности, тогда как компьютеры – нет. Само по себе, как мне кажется, это не может указать направление развития некой полезной научной теории интеллекта. Что уж такого особенного есть в биологических системах – если не принимать в расчет их «исторический» путь развития (и того, что мы оказались как раз такими системами), – что могло бы выделить их в качестве объектов, которым позволено «дорасти» до ментальности или семантики? Это заявление подозрительно напоминает мне догматическое утверждение, причем не менее догматического свойства, чем утверждения сторонников сильного ИИ о том, что, просто выполняя алгоритм, можно вызвать состояние осознанного восприятия!

По-моему, Серл, как и многие другие, были введены в заблуждение компьютерщиками. А тех, в свою очередь, сбили с толку физики. (Но это не вина физиков. Даже они не в состоянии

[89] См. статью Серла [1980], которая была опубликована в книге Хофштадтера и Деннетта [1981].

знать всё обо всем!) Вера в то, что «всё на свете является цифровыми компьютерами», кажется общераспространенной. И я намерен показать в этой книге, что это совсем не обязательно так.

* * *

В.Э.: Здесь я делаю обещанную вторую вставку с переводом фрагментов из моих ответов профессору Тамбергу (о «китайской комнате»).

(...)

§30. В китайской комнате

2000.05.19 13:36 пятница

.406. Эта одна глава из книги Пенроуза может послужить иллюстрацией вообще тому, каким образом проблемы интеллекта там обсуждаются. Аргументы о том, хватит ли Серлу времени, чтобы реализовать алгоритм Шенка в Китайской комнате, можно или нельзя привлекать жителей Индии...

.407. Всё это несущественная ерунда. Мы это оставим в стороне и возьмемся за сущность.

.408. Итак, ситуация следующая. Роджер Шенк[90] (1977) написал программу, которая «симулирует» понимание простеньких рассказиков («Съеден ли гамбургер?» и т.д.). Джон Серл (1980,[91] 1987)[92] выполняет «мысленный эксперимент»: он берет описание (на английском языке) алгоритма программы Шенка (для китайского языка), запирается в Китайской комнате, где через узкую щель ему подаются «истории» на китайском языке и потом кто-то таким же способом задает вопросы о них. Серл (Сирл? Сиерл?)[93], используя описание программы Шенка, выполняет все требующиеся манипуляции и возвращает назад нужный иероглиф, означающий «да» или «нет», и при этом он сам не понимает абсолютно ничего из того, что ему рассказывали, что спрашивали и что он ответил.

.409. Вот и вся сущность «Китайской комнаты». Вопросы о том, сколько Серлу потребуется времени, чтобы выполнить предусмотренные алгоритмом Шенка манипуляции, и можно ли в работу вовлекать индусов, здесь не имеют никакого значения. Важен только вопрос: КАКОВ алгоритм программы Шенка? Является ли это настоящей реализацией разума? – или это только обычная имитация? (Надо полагать, что реально Шенк в 1977 году написал типичную программу имитации). Если уж это имитация, то программа Шенка «не понимает», что она делает, и Серл, в свою очередь, тоже может не понимать ничего из того, что происходит.

.410. Однако обсуждать такие имитации неинтересно. Гораздо интереснее будет обсудить случай, когда вместо программы имитации находится настоящая программа интеллекта (например, не программа Шенка, а система Витоса, как она описана в главе о тесте Тьюринга {.315} и в других местах).

.411. Рассмотрим принципиальную схему, как должна действовать такая программа настоящего интеллекта, когда она читает рассказы на китайском языке, действительно понимает прочитанное, понимает заданные ей вопросы и осмысленно отвечает на них. (Будем использовать введенную в главе о тесте Тьюринга терминологию {.345} и называть такую программу настоящего интеллекта «Витосом»).

.412. В основных чертах (в блоках высшего уровня) алгоритм выполнения этих действий будет следующим:

.413. 1) Ввести в компьютер иероглифы «рассказа» и дешифровать их, то есть, констатировать принадлежность каждого знака к той или иной группе одинаковых знаков (в китайской письменности имеется примерно 6000 иероглифов, но так как для одного письменного знака «высшего уровня» могут использоваться комбинации нескольких иероглифов, то считается, что всего в китайском языке существуют около 50˙000 знаков; Витос, значит, должен в

---

[90] Schank, R.C. and Abelson, R.P. «Scripts, plans, goals and understanding». Eribaum, Hillsdale, NJ, 1977.

[91] Searle, J. «Minds, brains and programs», in «The behavioral and brain sciences», Vol.3. Cambridge University Press, 1980, reprinted in «The mind's I» (ed. D.R. Hofstadter and D.C. Dennett), Basic Books, Inc., Penguin Books Ltd., Harmondsworth, Middx. 1981.

[92] Searle, J.R. «Minds and brains without programs». In «Mindwaves» (ed. Blakemore and S. Greenfield), Basil Blackwell, Oxford, 1987.

[93] **В.Э.:** Читая книгу Пенроуза по-английски и пиша ответ Тамбергу по-латышски, я не знал, как произносится фамилия *Searle*.

первую очередь констатировать, которым из этих 50˙000 знаков является каждый очередной входящий знак; эту работу, конечно, осуществляет и программа Шенка, если она вообще может что-либо предпринять с китайскими «рассказами»).

.414. 2) «Понять», что означает каждый анализируемый знак, т.е. – найти в своей памяти объекты, связанные с данным знаком (нормально у человека с каждым словом понятного ему языка, например, с латышским «puķes» (цветы), связывается большое количество различных воспоминаний о виденных ранее объектах (напр., цветущие луга и т.д.), о читанных книгах (учебник по биологии и т.п.); эти объекты из своей памяти Витос должен активизировать если и не абсолютно все, то хотя бы часть из них; программы имитации же этой информации не имеют, эта работа ими не осуществляется, и именно это в первую очередь и отличает имитацию от подлинного понимания; ясно, что для того, чтобы такую связь между нововведенным иероглифом и ранее накопленными воспоминаниями было возможно установить, в памяти Витоса должна находиться эта ранее накопленная информация).

.415. 3) Основываясь на все эти образы (кадры), потоком входящих иероглифов активизированные, т.е. вызванные из ранее в памяти накопленной информации, конструировать в своей оперативной памяти картину о том, что именно высказано в читаемом «рассказе» (простоты ради мы эту конструированную «картину» можем представлять себе как изображение на экране дисплея, хотя картины, конструированные человеческим мозгом, могут включать не только визуальные образы, но и образы звуков, запахов и др.; конечно, программы имитации ничего подобного не делают, и именно это и означает, что они «не понимают» прочитанное).

.416. 4) Аналогичным образом расшифровать иероглифы заданного вопроса и (в принципе всё равно каким способом) кодировать в оперативной памяти сущность вопроса (способы этой кодировки можно спроектировать по-разному; у человека, вероятнее всего, вопросы кодируются в основном как альтернативные картины: одна картина, в которой гамбургер съедают, и другая, в которой его не едят, и т.д.)[94].

.417. 5) Основываясь на созданную в пункте (3) картину рассказа и созданную в пункте (4) кодировку вопроса, проверить ситуацию и генерировать нужный ответ («да» или «нет»).

.418. 6) В зависимости от сгенерированного в предыдущем пункте ответа, выбрать карточку с правильным иероглифом и засунуть ее в щель Китайской комнаты.

.419. Вот так действует настоящий интеллект, и та программа, которая осуществляет такой алгоритм, понимает, чтó она делает (помимо упомянутого здесь, Витос еще будет фиксировать в своей памяти выполненные им действия, чтобы позже можно было их в случае необходимости проверить и чтобы в следующих заданиях использовать эту информацию как для генерирования картинок, так и для формирования ответов и т.д.).

.420. Конечно, в названных выше шести блоках алгоритма показаны только самые «вершины айсберга»; каждый из этих блоков сложен сам по себе и интересен с точки зрения специалиста по информатике: как кодировать накопленную ранее в памяти информацию о цветах на лугах и об учебниках биологии?, как дешифрировать знаки иероглифов и определять, которым из 50˙000 знаков является введенный знак?, как установить связь между иероглифом и прежними воспоминаниями?, как из цепочки таких связок конструировать картину рассказа?, как проанализировать, удовлетворяет ли – или не удовлетворяет – эта картина условию, высказанному в вопросе?, как по результату этой проверки найти соответствующий иероглиф ответа?...

.421. Совсем не удивительно, что люди без опыта в программировании (в особенности в проектировании больших информационных систем) не могут всё это себе представить; точный ответ на все эти вопросы вообще будет зависеть также и от того, какова техническая конструкция самого компьютера, и от того, как сконструированы остальные (в прямом виде сюда не вовлеченные) аппараты этой операционной системы. Ну, а найти вообще принципиально возможные решения для меня, например, никаких трудностей не представляет.

.422. Пусть теперь Серл повторит алгоритм этой программы, как он это делал с программой Шенка. Если он это сможет, то, значит, он понял китайский язык. А если не поймет, то и не сможет повторить алгоритм Витоса.

.423. Серл не сможет повторить алгоритм Витоса, если в пункте (2) этого алгоритма ему придется использовать свои собственные воспоминания. Правда, мы можем здесь рассмотреть

---

[94] **В.Э.:** Плюс еще зачаток программы генерации ответа: объект (внутримозговой, внутрикомпьютерный), обрабатывая который, аппарат самопрограммирования создаст программы, выдающие ответ либо «да», либо «нет».

еще одну возможность: если Серл повторяет не только действия Витоса в Китайской комнате, но и вообще все его действия с самого начала, с самого его «рождения» – повторяет весь процесс накопления Витосом воспоминаний: когда ему впервые показывали, что такое цветок, произнося при этом название этого предмета на китайском языке, когда он читал книги по ботанике о цветах (иероглифами написанные) и т.д.

.424. Предположим, Серл проходит через всё это, но только накапливает он эти воспоминания не в собственной голове,[95] а на одной полке Китайской комнаты, в специально устроенной картотеке, имитируя все процессы этого накопления в таком же стиле, в каком он имитирует программу Шенка. Тогда, правда, одних жителей Индии ему будет недостаточно; надо будет звать на помощь не только всех людей Земли, но и обитателей нашей Галактики и других галактик, но, раз уж мы абстрагируемся от ресурсов времени и усилий, то это не существенно. Итак, допустим, что Серл повторил весь процесс накопления информации Витосом, сам не понимая, что он там накапливает в том углу Китайской комнаты.

.425. Тогда – да, – тогда он алгоритм витосовского интеллекта, читающий китайские иероглифы, повторить сможет, используя для этого не свой личный мозг, а то (внешнее для него) хранилище информации в углу комнаты.

.426. И что же тогда окажется произошедшим и что будет доказано? Тогда будет доказано, что Сирл <u>НЕ ЯВЛЯЕТСЯ</u> тем интеллектом, который понимает китайский язык. Китайским языком владеет <u>ТОТ</u> интеллект, который находится в Китайской комнате в целом: включая Серла (в роли простого интерпретатора), включая то хранилище информации в углу и включая ту распечатку программ Витоса, которой Серл руководствуется в своей работе имитации.

.427. Здесь мы соприкасаемся с тем вопросом, о котором размышляет и Пенроуз: что такое интеллект? Может ли им владеть механическая система шестерней? Или интеллект – это алгоритм? – и т.д.

.428. Конечно, интеллект присущ всякой системе, которая способна реализовать алгоритм Витоса в полном объеме и надлежащего качества. Если система шестерней способна это реализовать, то ей присущ интеллект (другое дело, конечно, что для реализации алгоритма Витоса необходима такая сложность, которую при помощи шестеренок никогда не достичь, но если бы достигли, – то интеллект присутствовал бы). Если этот алгоритм реализуется компьютером, то компьютеру присущ интеллект, если этот алгоритм реализует Серл вместе со шкафом в Китайской комнате, тогда у них (у обоих вместе взятых как у единой системы, а не у каждого в отдельности!) имеется интеллект (тот интеллект, который понимает китайский язык).

.429. Пенроуз усматривает «примечательную иронию» «в том факте», что сторонники таких взглядов якобы впадают в «крайнюю форму дуализма», ибо носителем интеллекта («*mind-stuff*») тогда получается алгоритм, существующий независимо от материи.

.430. А я тут никакого дуализма не могу увидеть. Интеллектуальной является та система, которая способна выполнить определенные, предварительно указанные действия. И всё.

.431. Является ли «носителем разума» алгоритм, или молекулы веществ этой системы, или еще что-нибудь – это всё только разглагольствования мыслителей типа Пенроуза, в какие я даже не пускаюсь: чтобы система могла осуществить те упомянутые «определенные, предварительно указанные» действия, необходимо всё – и молекулы веществ, и алгоритм, и память, и разные специфические блоки. Если не будет хотя бы одного компонента – не будет и интеллектуальной системы.

.432. Представлять интеллект (разум и т.д.) отделенным от той системы, которая его реализует, воображать, что, вот, здесь имеется система, а здесь, вот, ее разум, – такое мышление <u>УЖЕ</u> представляет собой неверный шаг. (И на самом деле именно сам Пенроуз никак не может освободиться от мышления такого рода, он всё время ищет, «где же сознание сидит?» – и, значит, он и есть тот дуалист; именно поэтому он и не может придти ко взглядам «*strong-AI*» – «строгого искусственного интеллекта»).

.433. Предположим, что кто-то о каком-нибудь теле, скажем, о танке, рассуждает так: «Вот, здесь танк, а здесь, вот, его тяжесть!». Два отдельных объекта – дуализм.

.434. Для меня отделять интеллект от системы столь же абсурдно, как отделять тяжесть от танка. Здесь вообще нет никакой проблемы, связанной с дуализмом; всё очень просто и даже

---

[95] **В.Э.:** Если он будет эти воспоминания накапливать в собственной голове, то, значит, научится китайскому языку; чтобы осталось в силе условие, что Серл китайским языком не владеет, воспоминания должны накапливаться вне его головы.

элементарно; проблему здесь могут видеть только те, кто никак не могут построить у себя в голове такую модель об этих вещах, какая имеется у меня (и, возможно, у других «*strong-AI*»), а всё время пользуются изначально дуалистической моделью, в которой разум всегда отделен от системы.

.435. Итак, интеллектом обладает всякая система, которая способна выполнить действия, определенные алгоритмом операционной системы Витос. Если Китайская комната в целом (включая Серла как интерпретатора и включая «шкаф внешней памяти» в углу) способна эти действия выполнить, то она обладает интеллектом (понимающим китайский язык). Интеллект собственно Серла (китайским языком не владеющий) в отношении к первому интеллекту выступает в такой же роли, как процессор компьютера в отношении к операционной системе: процессор тоже только «слепо» выполняет отдельные инструкции и сам по себе не умеет то, что умеет операционная система в целом.

.436. А если Серл «шкафом внешней памяти» не пользовался и не имитировал весь предыдущий рост интеллекта, то имитировать действия Витоса в Китайской комнате он не сможет.

.437. Как видите, решение проблемы «Китайской комнаты» столь же элементарно, как и решение проблемы континуума.

§31. Книга эйнштейновского мозга

.438. Раз уж Пенроуз в свою главу о Китайской комнате включил также пример Дугласа Хофстадера[96] (1981)[97] с «мозгом Эйнштейна», то рассмотрим и это заодно.

.439. Хофстадер придумал «книгу», в которой записана вся информация о мозге Альберта Эйнштейна («*a complete description of the brain*»), и Пенроуз, полемизируя с Хофстадером, обсуждает вопрос, была бы такая книга эквивалентна самому Эйнштейну, мыслила ли она, чувствовала бы, являлась ли «разумной» и т.д.

.440. Вся постановка вопроса здесь для специалиста по компьютерам представляется столь некорректной, что невзначай возникает вопрос: а что вообще и тот (Хофстадер), и другой (Пенроуз) знает о компьютерах?

.441. Мозг Эйнштейна (когда Эйнштейн жив и мозг работает) является компьютером, и в нем происходят какие-то информационные процессы (работают какие-то программы); эта работа называется мышлением, когда решаются какие-то большие (о единой теории поля) или малые (где находятся тапки?) вопросы; она называется чувствованием, когда из внешнего мира или изнутри организма приходят какие-то сигналы; она называется эмоциями, когда активизируются одни процессы и блокируются другие или же повышаются и понижаются пороги различных реакций в мозге – и т.д.

.442. Книга, напротив, никаким компьютером не является, и никакие информационные процессы в ней не происходят (тепловое движение молекул и подобные процессы во внимание принимать не будем). Для специалиста первый вопрос сразу возникает такой: <u>ЧТО</u> именно фиксировано в этой книге? «*A complete description of the brain*» для меня совершенно расплывчатое понятие, за которым могут скрываться совершенно разные вещи.

.443. Фиксировано ли в книге полностью одно моментальное состояние компьютера, т.е. вся находящаяся в его памяти информация (включая состояние программ) в какой-то один момент времени? Тогда эта книга похожа на широко известные «дампы» в области компьютеров.

.444. «Дампы» (копии памяти компьютера на какое-то заданное мгновение) отображают на каком-то другом носителе (например, на диске или на бумаге) всё состояние компьютера в какой-то определенный момент. Особенно широко этот способ использовался прежде, когда компьютеры еще были (относительно) небольшие и ненадежные. В 1970-х годах и ранее часто приходилось работать с программами, которые, чтобы решить какую-то задачу, должны были работать, скажем, сутки, а надежность самих ЭВМ была такой, что они «зависали» каждые несколько часов. Тогда, чтобы задачу вообще можно было решить до конца, через каждый определенный интервал времени (скажем, каждый час) делали на диске или магнитной ленте

---

[96] **В.Э.:** В русском переводе книги – Хофштадтер (очевидно подражая произношению его германских предков); но я пишу так, как это произносится в его родной Америке (они ведь произносят даже «Эйнстейн», а не «Эйнштейн», хотя Эйнштейн хотя бы родился в Германии и долго там жил).

[97] Hofstadter, D.R. «A conversation with Einstein's brain». In «The mind's I» (ed. D.R. Hofstadter and D.C. Dennett), Basic Books, inc.; Penguin Books, Ltd. Harmondsworth, Middx. 1981.

«дамп» – то есть полное изображение состояния компьютера. Если машина после этого «зависла» или «сдохла» (как программисты говорили на своем жаргоне), то из этого (последнего) «дампа» всю информацию загружали обратно в компьютер, и он продолжал работу с того места, на котором был записан «дамп». Таким образом можно было продолжать работу через аварии машины и в конце концов ее закончить, несмотря на ненадежность системы.

.445. «Дампы» на бумаге делались в моменты, когда выполнение программ (особенно операционных систем и других «системно-независимых» программ) завершалось «аварийно» (например, если в системе программ имелась ошибка). Тогда программист брал распечатанный дамп и, изучая состояние машины в момент аварии, искал ее причину. По такому дампу (если приложить достаточно усилий) действительно можно было многое сказать о том, что в машине происходило раньше, до аварии, или произошло бы потом (хотя по моментальному дампу и нельзя определить полностью всё, что имело место ранее или произошло бы позже).

.446. Итак, является ли «книга» Хофстадера чем-то похожим на такой «дамп»? Если так, то в какой момент времени дамп Эйнштейновского мозга делали? Или, может быть, там записаны все возможные дампы во все возможные моменты его жизни? Похоже, что ни Хофстадеру, ни Пенроузу такие вопросы даже не приходят в голову. А мне их дискуссия поэтому кажется чрезвычайно непрофессиональной и неконкретной.

.447. Так как говорится о том, что из этой книги якобы можно (в принципе) выяснить, что Эйнштейн ответил бы на тот или иной вопрос, то этим и будем руководствоваться. Тогда сформулируем и решим сами для себя такую проблему: можно ли каким-то способом дампировать состояние какого-то компьютера и определенной программной системы таким образом, чтобы по этому дампу (в принципе, игнорируя объемы требующейся работы) можно было определить всё, что эта программа делала бы в такой-то и такой-то ситуации?

.448. Здесь надо различать две вещи. Во-первых, по любому «обычному», т.е. записанному в какой-то конкретный момент, дампу можно сказать, чтó компьютер делал бы, если к нему в <u>этот момент</u> пришел бы какой-нибудь определенный сигнал (скажем, вопрос). Следовательно, если в «книге» Хофстадера в полном объеме фиксировано состояние мозга Эйнштейна в какое-то одно мгновение, то по ней (в принципе) можно было бы определить, что Эйнштейн (в этот момент!) ответил бы на тот или другой вопрос.

.449. Но, во-вторых, если мы хотим по этому (одномоментному) дампу знать, что Эйнштейн ответил бы через минуту, то мы должны знать также и то, какие еще импульсы пришли в мозг за эту минуту. В зависимости от этих импульсов ответ на тот же вопрос (сигнал, импульс) может быть совсем иным.

.450. Если в «книге» Хофстадера фиксированы и все те импульсы, которые пришли в мозг Эйнштейна на протяжении всей его жизни, то мы (в принципе) можем сказать, чтó он ответил бы в любой момент своей жизни (но тогда бессмысленным является вопрос, чтó Эйнштейн отвечает «вообще», как этот вопрос ставится в книге Пенроуза).

.451. Если в «книге» Хофстадера не фиксирован весь фактический ход жизни Эйнштейна, а о моментах времени, отличающихся от момента записи дампа, мы рассуждаем только «чисто теоретически» («..если бы до этого пришел бы такой-то импульс плюс еще такой-то..»), то число возможных комбинаций будет возрастать с чудовищной скоростью и, чем дальше от момента снятия дампа мы уйдем, тем более гипотетическими станут наши ответы о том, как реагировал бы Эйнштейн.

.452. Допустим теперь, что в «книге» Хофстадера «фактически реализуется» первый вариант: что в ней фиксирована вся жизнь Эйнштейна и все действительные состояния его мозга за время от 14 марта 1879 года до 18 апреля 1955 года так, что мы можем сказать, что он ответил бы в <u>любой</u> момент своей жизни на тот или иной вопрос.

.453. Но эквивалентна ли такая книга «в операциональном смысле» самому Эйнштейну (как это интерпретирует Пенроуз, чтобы потом отвергнуть)? Дамп (или собрание множества дампов) <u>никогда</u> не эквивалентен самому компьютеру. Почерпнуть информацию из дампа (и какие-то выводы о потенциальной реакции отображенного в дампе компьютера сделать) может только <u>другой</u> компьютер.

.454. В реальных бумажных дампах 1970-х годов этим другим компьютером был мозг программиста, который, руководствуясь фиксированной в дампе информацией, генерировал картины[98] действий «сдохшего» компьютера. Для книги Хофстадера этими другими компьютерами будут те «студенты», которые эту книгу читают и по ней создают для себя картины потенциальных ответов Эйнштейна. (Можно и заставить одного промышленного компьютера анализировать дамп другого промышленного компьютера и т.д.).

.455. Ясно, что книга Хофстадера является только носителем информации, и ничем более; ни о каком чувствовании или мышлении, или иной «эквивалентности» с самим Эйнштейном не может быть и речи.

.456. Однако, если мы в один «черный ящик» посадим самого Эйнштейна (допустим, что он еще жив), а во второй ящик – достаточно тренированного «студента» с книгой Хофстадера, – то по полученным ответам на основе теста Тьюринга невозможно будет отличить, в каком ящике сидит Эйнштейн, а в котором студент с книгой.

.457. Если мы имеем дамп одного компьютера, а сам этот компьютер сломался и ликвидирован, то мы можем этот дамп загрузить в другой компьютер такого же типа, и тот продолжит работу с того же места, на котором первый был отдампирован, – продолжит так, как будто и не было никакого перерыва в работе и никакя замена компьютеров не происходила.

.458. Поэтому, если мы имеем книгу Хофстадера и в добавок к ней еще и умение построить новое тело Эйнштейна, то из этой книги мы могли бы в это тело загрузить дамп соответствующего момента, – и Эйнштейн жил бы дальше, начиная с момента снятия соответствующего дампа. ЭТОТ Эйнштейн уже не будет простым носителем информации (как книга Хофстадера), а будет действительно во всех отношениях полностью эквивалентен «старому» Эйнштейну – он сможет и мыслить, и чувствовать, и переживать (конечно, если копирование тела и информации сделано достаточно качественно).

.459. Так как в книге Хофстадера (по нашему последнему предположению) фиксирована вся жизнь Эйнштейна (дампы во все ее моменты), то мы можем и создать большое количество Эйнштейнов разного возраста: каждый из них будет жить дальше с того момента, дамп которого в его мозг был загружен.

.460. Однако, если мы из этой книги Хофстадера одновременно запустим одного Эйнштейна (A) возраста 20 лет, и другого Эйнштейна (B) возраста 40 лет, то через 20 лет первый отнюдь не станет эквивалентен второму, каким он был в момент своего старта, так как те импульсы, та информация, которая за эти 20 лет войдет в Эйнштейна A, совсем не будет эквивалентной той информации, которую в промежутке между своими 20 и 40 годами получил самый первый Эйнштейн – тот, который жил в XIX и XX веках.

.461. Таковы вкратце профессиональные ответы на те вопросы, которые (непрофессионально) подняты Хофстадером и Пенроузом, – ответы, какими они получаются, если принять постулат, что человеческий мозг является дискретным биологическим компьютером. (Ну, если мозг не дискретный компьютер или вообще не компьютер, то, конечно, ответы получаются другими).

.462. Как я уже говорил, я не хочу уничижать Пенроуза – его заслуги в математической физике или «физической математике» неоспоримы –, но, как это было видно уже по этим нескольким примерчикам (и так оно во всей книге), его суждения в области информатики мне действительно представляются чрезвычайно некомпетентными: ну нет, нет у него понятия о возможной работе больших информационных систем, – и компетентные решения в его книге просто не рассматриваются.

(Конец вставки; далее продолжается текст Пенроуза)

* * *

§1.6. «Железо» и «софт»

На компьютерном жаргоне слово «железо» используется для обозначения всех устройств и элементов, из которых состоит компьютер (печатные платы, транзисторы, провода, накопители на магнитных дисках, и т.п.), включая также полное руководство по сборке. Аналогичным образом термин «софт» относится к различным программам, которые могут выполняться на

[98] **В.Э.:** Я использовал здесь слово «картины», не желая в этом тексте привлекать более точные, но читателю незнакомые термины, какие используются в Веданской теории. В принципе речь идет о структурах, отображающих объект и строящихся в мозге (или другом компьютере).

компьютере. Одним из замечательных открытий Тьюринга было то, что, по существу, любая машина с начинкой из «железа», характеризуемого определенной степенью сложности и гибкости, эквивалентна любой другой машине с такими параметрами. Эквивалентность двух машин (скажем, А и В) здесь должна пониматься в смысле точного соответствия действий А – при соответствующем заложенном в нее программном обеспечении – действиям В, и наоборот. Я употребляю здесь слово «точный» по отношению к конечным результатам, получающимся при введении в машины произвольных начальных данных (после того, как уже было введено преобразующее программное обеспечение), а не в смысле равенства времени, затраченного каждой машиной на получение ответа. Кроме этого, я допускаю для обеих машин возможность получения доступа к дополнительным (и, в принципе, неограниченным) внешним запасам чистых «черновиков» – магнитным пленкам, дискам, барабанам или иным носителям информации, – если какая-либо из них начинает испытывать нехватку в пространстве для хранения промежуточных результатов вычислений. Вообще говоря, разница между машинами А и В в затрачиваемом на выполнение некоторого задания времени может оказаться весьма серьезной. Вполне возможно, например, что машина А будет выполнять определенную задачу в тысячу раз быстрее, чем В. Равным образом может статься, что для другого задания время его выполнения машиной В окажется в тысячу раз меньше, чем машиной А. Более того, эти конкретные показатели могут в значительной степени зависеть от выбора используемых для конвертации программ. Но в рамках этой дискуссии нет нужды рассматривать такие практические аспекты, как способность выполнять вычисления за определенное время, поскольку наши рассуждения носят по большей части «принципиальный» характер. В следующем разделе я конкретизирую содержание тех концепций, которые затрагиваются здесь: машины А и В являют собой примеры того, что называют универсальными машинами Тьюринга.

В сущности, все современные общеупотребительные компьютеры – это универсальные машины Тьюринга. Тем самым все такие компьютеры будут эквивалентны друг другу в вышеупомянутом смысле: различия между ними будут заключаться единственно в программном обеспечении, при условии, что нас не волнует разница в скорости выполнения операции и возможные ограничения пространства для хранения данных. Но современные технологии сделали компьютеры способными работать так быстро и с такими огромными объемами памяти, что для большей части «повседневных» задач ни один из этих практических аспектов не накладывает серьезных ограничений на спектр решаемых такими компьютерами задач[99] – так что эта эффективная эквивалентность, введенная на теоретическом уровне, просматривается и на практике. Кажется, что технология превратила совершенно абстрактные когда-то академические дискуссии об идеальных вычислительных устройствах – в устройства реальные, и непосредственно влияющие на нашу жизнь!

Насколько я могу понять, одним из наиболее важных положений, на которых базируется философия сильного ИИ, является именно эта эквивалентность между различными физическими вычислительными устройствами. «Железо» расценивается как сравнительно (или вообще) несущественный фактор, в то время как «софт», т.е. программа или алгоритм, считается единственным жизненно важным компонентом. Однако, мне кажется, что существуют и другие, не менее важные «краеугольные камни здания сильного ИИ», которые следуют из физики. Сейчас я попытаюсь дать некоторое представление об их природе.

Что позволяет нам идентифицировать себя как личность? Может быть, в какой-то степени – сами атомы наших тел? Особые сочетания электронов, протонов и других частиц, из которых состоят эти атомы? Есть, по крайней мере, два возражения против этого предположения. Во-первых, вещество тела любого живого существа претерпевает постоянные изменения и обновления. Это справедливо, в частности, для клеток головного мозга, несмотря на то, что после рождения новые клетки уже не образуются. Абсолютное большинство атомов в каждой живой клетке (включая все клетки мозга) – и, конечно же, практически все ткани нашего тела – замещаются новыми по много раз с момента рождения.

Второе возражение приходит из квантовой физики – и, по странной иронии, находится, строго говоря, в прямом противоречии с первым! Согласно квантовой механике (и мы узнаем об этом больше в главе 6, {с.226}) любые два электрона должны быть с необходимостью одинаковыми; и то же самое справедливо в отношении двух произвольно взятых протонов или пары любых других частиц, относящихся к одному типу. То, что подразумевается под этим,

---

[99] См., однако, рассуждения о теории сложности и NP-задачах в конце главы 4.

отнюдь не ограничивается утверждением об их неразличимости – оно значительно сильнее. Если пришлось бы поменять между собой электрон в человеческом мозге и электрон в кирпиче, то состояние системы осталось бы в точности тем же самым,[100] что и до этого – тем же самым, а не просто неотличимым! Аналогичное правило справедливо и для протонов, и для других разновидностей частиц, а также для целых атомов, молекул и т.п. Если весь материал человеческого тела заместить соответствующими частицами кирпичей из его дома, то, в буквальном смысле, вообще ничего не изменится. То, что отличает человека от своего дома – это то, в какую структуру организованы составляющие его тела, а не индивидуальные свойства этих составляющих.

Можно привести аналогию из повседневной жизни, не имеющую отношения к квантовой механике, которая бросилась мне в глаза, пока я набирал эти строки, имея в своем распоряжении один из плодов информационной технологии – текстовый редактор. Если я хочу изменить слово, скажем, «болт» на «борт»[101], то могу сделать это просто заменив букву «л» буквой «р»; или же я могу вместо этого напечатать всё слово заново. Выбрав последний вариант, я встану перед вопросом: а та ли это теперь буква «б», что была ранее, или я заменил ее идентичной? А как насчет «т»? Даже если я решу просто поменять букву «л» на «р», а не перебивать всё слово заново – будет момент, как раз между удалением «л» и появлением «р», когда пустое место «схлопывается»[102] и по всему тексту сверху вниз пройдет волна перестановок, при которых пересчитывается расположение всех букв, включая «т» – а затем пере-пересчитывается еще раз при вставке на то же место «р». (Ох уж эта дешевизна бездумных вычислений в наши дни!) В любом случае, все буквы, которые я вижу на экране, есть не более чем разрывы на пути следования электронного луча в процессе сканирования всего экрана, происходящего шестьдесят раз в секунду. Если я возьму произвольную букву и заменю ее на такую же – сохранится ли при этом исходное состояние <u>точно таким же</u> или оно будет только лишь неотличимо? Попытка провести смысловое разделение между двумя этими определениями нового состояния (т.е. между «только лишь неотличимое» и «точно такое же») кажется несерьезной. По крайней мере, коль скоро замещающая буква является идентичной, возникает желание назвать это состояние таким же. И то же самое верно и для квантовой механики одинаковых частиц. Поменять одну из частиц на другую, эквивалентную – всё равно, что не поменять ничего. Состояние при этом должно считаться <u>тем же самым</u>, что и в начале. (Однако, как станет ясно в главе 6, подобное различие <u>не так уж</u> тривиально в контексте квантовой механики.)

Рассуждения, сделанные выше по поводу непрерывного обновления атомов человеческого тела, надо рассматривать скорее в рамках классической физики, нежели квантовой. В этих рассуждениях используется терминология, которая неявно подразумевает возможность индивидуального существования каждого атома. На этом уровне описания классическая физика вполне адекватна и мы не слишком погрешим против истины, если будем рассматривать атомы в качестве отдельных объектов. При условии, что атомы достаточно хорошо отделены друг от друга в процессе движения, <u>можно</u> было бы говорить об их индивидуальном существовании, поскольку каждый атом допускает в этом случае непрерывное наблюдение за собой. С точки зрения квантовой механики говорить об индивидуальности атомов можно только ради удобства описания, однако на рассматриваемом уровне это вполне допустимо.

Давайте примем, что индивидуальность человека никак не связана с индивидуальностью, которую можно было бы постараться приписать его материальной основе. Вместо этого она должна определяться своего рода <u>конфигурацией</u> составляющих элементов этой основы – их пространственной или, допустим, пространственно-временно́й структурой. (Подробно об этом – далее.) Но сторонники сильного ИИ идут еще дальше. Если информационное содержание такой конфигурации перевести в другую форму, из которой затем можно было бы полностью восстановить оригинал, то, согласно их утверждению, индивидуальность человека осталась бы неизменной. Это похоже на ситуацию с последовательностью букв, которую я только что напечатал и теперь вижу на дисплее моего текстового редактора. Если я уберу их с экрана, то они, тем не менее, сохранятся записанными в виде определенных крошечных изменений

---

[100] Некоторые читатели, сведущие в этом вопросе, могли бы возмутиться из-за некоторой разницы в знаках. Но даже это (спорное) различие пропадет, если мы при замене повернем один из электронов на 360 градусов! (Пояснения можно найти на с. 226, глава 6.)

[101] **В.Э.:** В оригинале: *to transform `make' into `made'.*

[102] **В.Э.:** Пенроуз, видимо, работал с редактором типа *Word*; в редакторе, например, МЕ при режиме OVR это будет не так.

электрического заряда, в конфигурации, геометрически никак не соотносящейся с буквами, которые я минуту назад напечатал. И всё же в любой момент я могу вернуть их на экран – и вот они, пожалуйста, точь-в-точь такие же, словно и не было никаких преобразований. Если я захочу сохранить написанное, то я могу перевести информацию о последовательности букв в некоторую конфигурацию намагниченных доменов[103] на диске, который я затем выну и выключу машину, аннулируя тем самым все (соответствующие) крошечные изменения заряда в ячейках ее памяти. Тогда завтра я смогу снова вставить диск,[104] восстановить эти смещения и отобразить последовательность букв на экране так, как будто ничего и не случилось. Приверженцам теории сильного ИИ «ясно», что аналогичным образом можно обращаться и с личностью человека. Как и в случае с буквами у меня на экране, скажут они, человеческая индивидуальность ничего не потеряла бы – собственно, с ней вообще ничего бы не произошло, – если ее физическую форму перевести во что-нибудь совершенно иное, скажем, в поля намагниченности железного бруска. Они, кажется, даже готовы поспорить, что сознательное восприятие человека сохранилось бы и в то время, пока «информация» о нем пребывает в другой форме. При таком подходе «человеческое сознание» должно рассматриваться, по сути, как набор программ – «софта»[105], – а его конкретное воплощение в виде материального человеческого существа – как действия этих программ, осуществляемые «железной начинкой» его тела и мозга.

Основанием для подобных заявлений служит, вероятно, убежденность в том, что какую бы материальную форму не принимало «железо» – пусть это будет, например, какое-нибудь электронное устройство, – ему можно будет всегда «задать» вопрос-программу (в духе теста Тьюринга), и ответ на него, в предположении о способности «железа» адекватно вычислять ответы на эти вопросы, будет неотличим от ответа человека, данного им в нормальном психическом состоянии. («Как вы чувствуете себя сегодня утром?» – «О, вполне сносно, хотя мне немного докучает легкая головная боль». – «Значит, вы не чувствуете... э-э... ну, чего-нибудь необычного, связанного с вашей личностью... ничего такого?» – «Нет. А почему вы спрашиваете об этом? Довольно странный, знаете ли, вопрос...» – «То есть вы чувствуете себя тем же самым человеком, что и вчера?» – «Ну конечно!»[106])

Идея, которую часто обсуждают в связи с этим, носит в фантастической литературе название <u>телепортационной машины</u>.[107] Предполагается использовать ее для транспортировки, допустим, с одной планеты на другую; но будет ли она работать именно таким образом – это как раз и является предметом обсуждения. Вместо того, чтобы перемещаться «обычным» путем – на космическом корабле, – гипотетический путешественник подвергается сканированию с макушки до пят, при котором со всей возможной аккуратностью фиксируется положение и характеристики каждого атома в его теле. Затем вся эта информация передается со скоростью света при помощи любого подходящего электромагнитного сигнала на ту планету, где он хотел бы оказаться. Там эта информация собирается воедино и используется в качестве инструкций для создания точной копии путешественника, со всеми его воспоминаниями, устремлениями, надеждами и самыми глубокими чувствами. По крайней мере, так это должно выглядеть на практике: все детали состояния мозга подробно записываются, затем передаются, и по этим данным происходит реконструирование. Если предположить, что всё произошло так, как надо, то оригинал можно «безболезненно» уничтожить. В таком случае возникает вопрос: является ли такой механизм <u>настоящим</u> путешествием с одного места на другое – или же это просто создание дубликата, сопровождающееся убийством оригинала? Будете ли вы готовы воспользоваться таким способом

---

[103] **В.Э.:** По-русски лучше – участков.

[104] **В.Э.:** Речь, очевидно, идет о дискетах. Интересно – у Пенроуза, видимо, тогда (в 1980-х) не было твердого диска?

[105] **В.Э.:** Плюс накопленная информация.

[106] **В.Э.:** Этот диалог по замыслу Пенроуза, видимо, происходит после того, как «человеческую личность» пересадили из биологического тела в электронное устройство. Но «пересаженная личность» не почувствует никаких изменений (и будет отвечать так, как пишет Пенроуз) только в том случае, если для нее будет полностью имитирован весь окружающий мир и поступающие от него сигналы, и имитирован таким, каким он его ожидает видеть по своему предыдущему опыту. Если такую задачу имитации внешнего мира (включая собственное тело) успешно выполнят, то – да, субъект будет отвечать так, как в диалоге Пенроуза. А если не выполнят, то, конечно, на субъекта обрушатся совершенно новые сигналы, требующие такой обработки, для которой у него нет никаких заготовок. (Или Пенроуз не собирается посылать в пересаженный интеллект вообще никаких сигналов о внешнем мире и собственном теле?)

[107] См. вступление к книге Хофштадтера и Деннетта [1981].

«путешествия» при условии, что он подтвердит свою стопроцентную надежность? Если телепортация не является путешествием, то в чем же заключается принципиальная разница между ней и простым переходом из одной комнаты в другую? А в последнем случае – разве не определяют атомы в один момент времени информацию об их положении в последующие моменты? В конце концов, мы видели, что сохранять «индивидуальность» какого бы то ни было атома – нецелесообразно. Вопрос об индивидуальных характеристиках атома вообще не имеет смысла. Разве произвольная движущаяся структура из атомов не представляет собой своего рода волну информации, распространяющуюся между точками пространства? Тогда есть ли существенная разница между распространением волн, несущих информацию о переходящем из комнаты в комнату человеке, – и тех, что посылаются устройством телепортации?

Допустим, что телепортация действительно «работает» в том смысле, что «сознание» путешественника на самом деле просыпается в его двойнике, находящемся на далекой планете. Что тогда произойдет в том случае, если мы, в нарушение правил игры, не уничтожим оригинал путешественника? Будет ли его «сознание» одновременно в двух разных местах[108]? (Попытайтесь представить свою реакцию на следующее заявление: «Ах, дорогой, похоже, суспензия, которую мы дали тебе перед посадкой в Телепортатор, испортилась раньше срока? Да, вышло не очень удачно, хотя это не так страшно. В любом случае, тебе, наверное, будет приятно услышать, что другой ты – ну-у, то есть, конечно, настоящий ты – прибыл на Венеру в целости и сохранности, поэтому мы можем... э-э... избавиться от тебя здесь – нет, я имею виду... ну, от ненужной больше копии. Разумеется, это пройдет совершенно безболезненно».) Возникает парадоксальная ситуация. Существуют ли в физике законы, делающие телепортацию принципиально невозможной? С другой стороны, возможно, там нет никаких абсолютных запретов на такую «передачу» человека и его сознания, но сам принцип «копирования» предполагает неизбежное уничтожение оригинала[109]? Может быть, сохранение двух дееспособных копий запрещено в принципе? Хотя эти рассуждения носят отстраненный характер, я всё же верю, что из них можно извлечь кое-какие полезные сведения о физической природе сознания и индивидуальности. Я вижу в них явное указание на ту существенную роль, которую играет квантовая механика в понимании явлений умственной деятельности. Но я слишком забегаю вперед. К этой теме необходимо будет вернуться после того, как мы изучим структуру квантовой теории в главе 6 (см. {с.220}).

Давайте посмотрим, какое отношение имеет теория сильного ИИ к вопросу о телепортации. Мы предположим, что где-то между двумя планетами располагается ретрансляционная станция, на которой полученная информация некоторое время хранится перед тем, как быть отправленной к месту своего назначения. Для удобства эта информация записывается не в человеческой форме, а в каком-нибудь электронном или магнитном устройстве. Будет ли человеческое «сознание» присутствовать в этом устройстве? Приверженцы сильного ИИ постарались бы убедить вас в том, что это будет именно так.[110] Ведь в конечном счете, сказали бы они вам, на любой вопрос, который мы решили бы задать путешественнику, могло бы, в принципе, ответить и это устройство – если «просто» сымитировать соответствующую функцию его мозга.[111] Устройство располагало бы всей необходимой информацией, и дело стало бы только за вычислениями. А если устройство отвечает на вопросы в точности так же, как если бы это был путешественник, то (с точки зрения теста Тьюринга!) оно им и является. В качестве основы для

---

[108] **В.Э.:** Разумеется, будет, то есть, будут два субъекта с одинаковой стартовой точкой – но с неодинаковыми дальнейшими путями. (Эту ситуацию я обыграл в своем юношеском романе «Робинзоны Буэноса», написанном в 1963 году, когда мне было 16 лет). Вообще все эти вопросы еще один лишний раз показывают, сколь ненадежной системой на самом деле является «человеческая личность» («..на соплях держится..»). Даже и без этих дел тому есть огромное количество доказательств. Природа (естественный отбор) создал канву для «естественного» развития этой личности, и пока она в этой канве, она кажется более менее «надежной», но как только выйти за пределы, установленные естественным отбором...

[109] **В.Э.:** Нет, не предполагает. Компьютерные системы же дают нам образец того, как всё обстоит на самом деле. Что можно делать с компьютерными системами, то можно (в принципе) и с человеческой личностью. (Так вытекает из постулата о том, что человеческая психика – лишь система обработки информации в организме; сам этот постулат уже достаточно обсуждался, и не будем здесь всё это повторять).

[110] **В.Э.:** Нет. Это будет просто «дамп» компьютера, статус которого я уже описал в последней вставке в текст Пенроуза.

[111] **В.Э.:** «Дамп» не ответил бы; ответить мог бы тот компьютер, который анализирует этот дамп.

такого вывода здесь опять выступает известное утверждение сторонников сильного ИИ: для явлений, связанных с умственной деятельностью, «железо» не имеет никакого значения. Это утверждение кажется мне неправомочным. Оно, в свою очередь, основывается на представлении о мозге (или разуме) как о цифровом компьютере. И подразумевает, что нет каких-то особых физических процессов, приводящихся в действие, когда человек думает,[112] которые могли бы требовать для своей реализации ту конкретную физическую (биологическую, химическую) структуру, которой обладает мозг.

Естественно, проповедники сильного ИИ будут настаивать на том, что единственное предположение, которое при этом вводится, касается универсальной возможности численного моделирования[113] любого физического процесса. Я более чем уверен, что подавляющее большинство физиков, опираясь на современное состояние физической науки, сочло бы такое предположение совершенно оправданным. В следующих главах я представлю свои собственные доводы в пользу противоположной точки зрения (а также подготовлю почву, чтобы объяснить, почему я думаю, что делается некое предположение). Но давайте на мгновение примем (широко распространенную) точку зрения, согласно которой все относящиеся к предмету дискуссии физические процессы допускают численное моделирование. Тогда единственным (если не принимать во внимание вопросы о времени и ресурсах, затраченных на вычисления) реальным предположением будет следующее «операционалистское» предположение: если нечто действует в точности, как существо, обладающее осознанным восприятием, то мы должны считать, что оно себя этим существом и «чувствует».

Точка зрения теории сильного ИИ состоит в том, что, рассматривая «только» вопрос, относящийся к «железу», любые физические процессы, имеющие отношение к работе мозга, в обязательном порядке могут быть промоделированы с помощью соответствующего преобразующего «софта». Если мы принимаем операционалистскую точку зрения, то тогда этот вопрос будет состоять в эквивалентности универсальных машин Тьюринга, в том, что такие машины способны выполнять любой алгоритм, – а также в справедливости предположения об алгоритмической природе деятельности мозга. И теперь самое время коснуться этих интригующих и важных понятий более подробно.

Глава 2. Алгоритмы и машины Тьюринга

§2.1. Основы алгоритмов

Как точно определить понятие алгоритма, или машины Тьюринга, или универсальной машины Тьюринга? Почему эти понятия играют одну из главных ролей в современном представлении о «мыслящем устройстве»? Есть ли какие-нибудь абсолютные ограничения на принципиальные возможности использования алгоритмов? Для того, чтобы ответить на эти вопросы, нам придется разобраться в деталях, что представляют собой алгоритм и машины Тьюринга.

В дальнейших рассуждениях я буду иногда прибегать к математическим выражениям. Вероятно, некоторых читателей эти выкладки напугают и даже заставят отложить книгу в сторону. Если вы как раз такой читатель, то я прошу вашего снисхождения и рекомендую вам последовать совету, данному мной в *Обращении к читателю* на с. 6! Доказательства, которые здесь встретятся, не потребуют владения математическим аппаратом, выходящим за пределы школьного курса, но чтобы в них детально разобраться, всё же понадобятся интеллектуальные усилия. На самом деле, большинство рассуждений изложено весьма подробно, и если внимательно им следовать, можно добиться глубокого понимания. Однако, даже беглый просмотр

---

[112] **В.Э.:** Это вопрос постулата; можно принять (в нашей строящейся модели о сущности разума) тот или иной постулат, но проще та система постулатов, которая не предполагает никаких «особых физических процессов», и если в ней всё можно объяснить (а мне представляется, что можно), тогда и нет необходимости принимать какие-то дополнительные постулаты.

[113] **В.Э.:** Далось Пенроузу это «численное моделирование»! Не о моделировании речь, а о том, что информацию могут обрабатывать различные системы – и биологические компьютеры типа мозга, и электронные компьютеры типа теперешних промышленных. Различные устройства могут делать одинаковую работу – вот корень дела.

доказательств позволяет ухватить основную идею. С другой стороны, если вы являетесь экспертом в этой области, то я опять вынужден принести свои извинения. Но я осмелюсь предположить, что даже в этом случае вам будет небесполезно ознакомиться с моими рассуждениями, в которых почти наверняка найдется что-то интересное и для вас.

Слово «алгоритм» происходит от имени персидского математика IX века Абу Джафара Мухаммеда ибн Мусы аль-Хорезми, написавшего около 825 года н.э. руководство по математике *Kitab al-jabr wa'l-muqabala*, которое оказало значительное влияние на математическую мысль того времени. Современное написание «алгоритм», пришедшее на смену более раннему и точному «алгоризм»[114], своим происхождением обязано, скорее всего, ассоциации со словом «арифметика»[115]). (Примечательно, что и слово «алгебра» происходит от арабского *al-jabr,* фигурирующего в названии вышеупомянутой книги.)

Примеры алгоритмов были, однако, известны задолго до появления книги аль-Хорезми. Один из наиболее известных – алгоритм Евклида – процедура отыскания наибольшего общего делителя двух чисел, восходит к античности (примерно 300 лет до н.э.). Давайте посмотрим, как он работает. Возьмем для определенности два числа, скажем, 1365 и 3654. Наибольшим общим делителем двух чисел называется самое большое натуральное число, на которое делится каждое из этих чисел без остатка. Алгоритм Евклида состоит в следующем. Мы берем одно из этих чисел, делим его на другое и вычисляем остаток: так как 1365 входит дважды в 3654, в остатке получается 3654 – 2 × 1365 = 924. Далее мы заменяем наши два исходные числа делителем (1365) и полученным остатком (924), соответственно, производим с этой парой ту же самую операцию и получаем новый остаток: 1365 – 924 = 441. Для новой пары чисел – а именно, 924 и 441, – получаем остаток 42. Эту процедуру надо повторять до тех пор, пока очередная пара чисел не поделится нацело. Выпишем эту последовательность:

3654 : 1365	дает в остатке 924
1365 : 924	дает в остатке 441
924 : 441	дает в остатке 42
441 : 42	дает в остатке 21
42 : 21	дает в остатке 0.

Последнее число, на которое мы делим, а именно 21, и есть искомый наибольший общий делитель.

Алгоритм Евклида является <u>систематической процедурой</u>, которая позволяет найти этот делитель. Мы только что применили эту процедуру к двум конкретным числам, но она работает и в самом общем случае с произвольными числами. Для очень больших чисел эта процедура может занять много времени, и будет выполняться тем дольше, чем больше сами числа. Но в каждом конкретном случае выполнение процедуры в конце концов заканчивается, приводя за конечное число шагов к вполне определенному ответу. На каждом этапе мы точно представляем себе действие, которое должно быть выполнено, и точно знаем, когда получен окончательный результат. Более того, всю процедуру можно описать <u>конечным</u> числом терминов, несмотря на то, что она может применяться к любым, сколь угодно большим натуральным числам. («Натуральными числами» называются неотрицательные[116] целые числа 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...) На самом деле нетрудно изобразить (конечную) блок-схему, описывающую логическую последовательность операций алгоритма Евклида (рис. 2.1).

Нужно заметить, что на схеме эта процедура не до конца разбита на простейшие составляющие, поскольку мы неявным образом предположили, что нам уже «известно», как выполнять необходимую базовую операцию получения остатка от деления двух произвольных натуральных чисел A и B. Эта операция, в свою очередь, также алгоритмична и выполняется при помощи хорошо знакомой нам со школы процедуры деления. Эта процедура, на самом деле, сложнее, чем

---

[114] **В.Э.:** В оригинале: *algorism*.

[115] Автор имеет в виду созвучность английских слов *algorithm* и *arithmetic*. – *Прим. ред.* – **<u>В.Э.</u>:** Вовсе нет. Автор имеет в виду то, что в латинском (а от него – и в английском) языке в этих словах еще в средние века появилось одинаковое написание частицы «*rithm*», хотя происхождение слов совершенно разное: первое слово – это искаженное арабское «*Al Horezmi*» («из Хорезма»), а второе – греческое «*arithmos*» («число») + окончание, означающее науку. Так вот, Пенроуз говорит, что искажение слова «*Horezm*» до «*gorithm*» произошло под влиянием слова «*arithm*» (какие-то средневековые авторы или переписчики не поняли и перепутали; первоначальное «*algorism*» как раз и стоит ближе к «*Al Horezm*»).

[116] Я использую обычную современную терминологию, в которой множество «натуральных чисел» включает и нуль.

все остальные части алгоритма Евклида, но и она может быть представлена в виде блок-схемы. Основное затруднение здесь возникает из-за использования привычной «десятичной» записи натуральных чисел, что вынуждает нас выписывать все таблицы умножения, учитывать перенос и т.п. Если бы для представления некоторого числа n мы использовали последовательность из n каких-нибудь одинаковых знаков, например, пяти звездочек (*****) для обозначения пятерки, то определение остатка свелось бы к совершенно элементарной алгоритмической операции. Для того, чтобы получить остаток от деления А на В, достаточно просто убирать из записи числа А последовательность знаков, представляющих В, до тех пор, пока на некотором этапе оставшееся число знаков в записи А не станет недостаточным для выполнения следующего шага. Эта последовательность знаков и даст требуемый ответ. Например, желая получить остаток от деления 17 на 5, мы просто будем последовательно удалять ***** из *****************, как это показано ниже:

```
*****************
     ************
          *******
               **,
```

и в результате получим, очевидно, 2, так как следующее удаление уже станет невозможно.

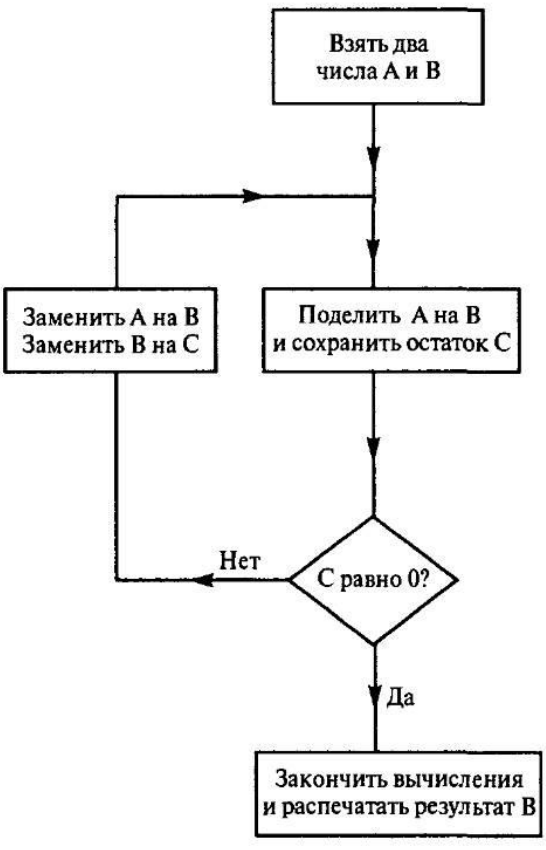


Рис. 2.1.

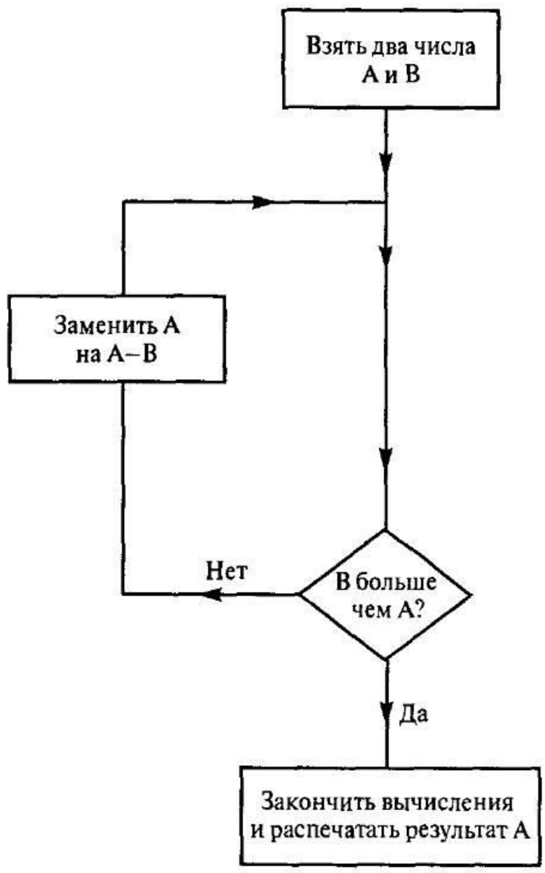


Рис. 2.2.

Блок-схема изложенного выше процесса нахождения остатка от деления путем последовательных вычитаний приведена на рис. 2.2. Чтобы придать блок-схеме алгоритма Евклида завершенный вид, мы должны подставить схему отыскания остатка в соответствующий блок справа в центре предыдущей схемы. Такая подстановка одного алгоритма в другой – распространенная в компьютерном программировании процедура. Алгоритм вычисления остатка, изображенный на рис. 2.2, служит примером <u>подпрограммы</u>, иначе говоря, это алгоритм (как правило, уже известный), вызываемый и используемый по мере надобности в ходе выполнения основного алгоритма.

Безусловно, обозначение числа n просто набором из n звездочек чрезвычайно неэффективно, когда речь заходит о больших числах. Именно поэтому обычно используют более компактную запись, например, стандартную (десятичную) систему. Однако оставим в стороне эффективность операций и обозначений и уделим всё внимание вопросу о том, какие операции в принципе могут выполняться алгоритмически. Действие, которое поддается алгоритмизации в одной записи, сохранит это свойство и в любой другой. Эти два случая различаются только техническими нюансами и сложностью выполнения алгоритма.

Алгоритм Евклида – это лишь одна из многих, часто классических, алгоритмических процедур, встречающихся в математике повсеместно. Но, вероятно, не лишним будет отметить, что, несмотря на значительный исторический возраст отдельных алгоритмов, точная формулировка универсального определения алгоритма появилась только в двадцатом веке. В 1930-х годах было предложено несколько альтернативных формулировок этого понятия, из которых наиболее емкая и убедительная – и, к тому же, наиболее значимая в историческом плане – опирается на понятие машины Тьюринга. Поэтому нам будет полезно рассмотреть некоторые свойства этих «машин».[117]

Прежде всего следует помнить, что «машина» Тьюринга принадлежит области «абстрактной математики» и ни в коем случае не является физическим объектом. Это понятие было введено в 1935–1936 годах английским математиком и кибернетиком Аланом Тьюрингом, внесшим огромный новаторский вклад в развитие компьютерной науки (Тьюринг [1937]). Тьюринг рассматривал задачу весьма общего характера (известную как проблема алгоритмической разрешимости), которая была поставлена великим немецким математиком Давидом Гильбертом частично в 1900 году на Парижском Конгрессе математиков (так называемая «десятая проблема Гильберта»), и более полно – на международном конгрессе 1928 года в Болонье. Проблема, поставленная Гильбертом, состояла ни больше, ни меньше как в отыскании универсальной алгоритмической процедуры для решения математических задач или, вернее, ответа на вопрос о принципиальной возможности такой процедуры. Кроме того, Гильберт сформулировал программу, целью которой было построение математики на несокрушимом фундаменте из аксиом и правил вывода, установленных раз и навсегда. Но к тому моменту, когда Тьюринг написал свою великую работу, сама идея этой программы уже была опровергнута поразительной теоремой, доказанной в 1931 году блестящим австрийским логиком Куртом Гёделем. Мы рассмотрим теорему Гёделя и ее значение в четвертой главе. Проблема Гильберта, которую исследовал Тьюринг (*Entscheidungsproblem*), не зависит от какого-либо конкретного построения математики в терминах аксиоматической системы. Вопрос формулировался так: существует ли некая универсальная механическая процедура, позволяющая, в принципе, решить все математические задачи (из некоторого вполне определенного класса) одну за другой?

Трудность с ответом на этот вопрос была связана отчасти с определением смысла «механической процедуры» – это понятие выходило за рамки стандартных математических идей

---

[117] **В.Э.:** Тьюринг создавал свою «машину», когда в мире еще не было компьютеров, – но спустя примерно десятилетие они появились. С тех пор выросли уже несколько поколений компьютерных программистов, которые занимались этим делом профессионально – и всю жизнь (к ним принадлежу и я). И нам понятия «программа» и «алгоритм» были самыми что ни есть основными рабочими понятиями. Но наше, программистское, понимание алгоритма отличается от тьюрингова – может не столько по существу, сколько по расстановке акцентов. Для нас несущественно, что «алгоритм – это механическая, систематическая процедура», которую нужно расписать до последнего шага и тем самым примитизировать, как это будет делаться в (описываемых Пенроузом) «машинах Тьюринга». Для нас существенно другое. «Программа» – это объект конкретный: «вот, эта программа сейчас загружена в этот компьютер и работает». А «алгоритм» – это объект абстрактный: это принцип, по которому программа работает (и один алгоритм может быть осуществлен, встроен во многих разных программах). (Так слова «программа» и «алгоритм» нужно понимать во всех моих текстах, в том числе тех, что описывают Веданскую теорию). И чтобы понять этот принцип работы программы (т.е. ее алгоритм), отнюдь не надо стараться расписать каждый мелкий шаг и представлять элементарные действия типа движения ленты вправо и влево, записи и стирания ноликов и единиц на ней. Наоборот, такие представления могут только мешать пониманию алгоритма. (Поэтому мы, программисты, и не любим «машины Тьюринга» и не ценим их высоко). Понимание алгоритма должно начинаться «сверху», а не «снизу», как у Тьюринга. Сначала понимаешь общие блоки происходящего, потом спускаешься к более мелким блокам, потом еще дальше к деталям... А дергающуюся туда–сюда ленту вообще незачем представлять. Блок-схемы тоже могут изобразить только очень примитивные алгоритмы. Настоящие программисты их никогда не рисуют (разве что только при обучении девушек-практиканток).

того времени. Чтобы как-то ее преодолеть, Тьюринг постарался представить, как можно было бы формализовать понятие «машина» путем расчленения ее действий на элементарные операции. Вполне вероятно, что в качестве примера «машины», помимо прочего, Тьюринг рассматривал и человеческий мозг, тем самым относя к «механическим процедурам» все действия, которые математики выполняют, размышляя над решением математических задач.

Хотя такой взгляд на процесс мышления оказался весьма полезным при разработке Тьюрингом его в высшей степени важной теории, нам совершенно необязательно его придерживаться. Действительно, дав точное определение механической процедуры, Тьюринг тем самым показал, что существуют совершенно четко определенные математические операции, которые никак не могут называться механическими в общепринятом смысле слова. Можно, наверное, усмотреть некую иронию в том, что эта сторона работы Тьюринга позволяет нам теперь косвенным образом выявить его собственную точку зрения на природу мышления. Однако, нас это пока занимать не будет. Прежде всего нам необходимо выяснить, в чем же, собственно, заключается теория Тьюринга.

§2.2. Концепция Тьюринга

Попробуем представить себе устройство, предназначенное для выполнения некоторой (конечноопределенной) вычислительной процедуры. Каким могло бы быть такое устройство в общем случае? Мы должны быть готовы к некоторой идеализации и не должны обращать внимания на практические аспекты – мы на самом деле рассматриваем математическую идеализацию «машины». Нам нужно устройство, способное принимать дискретное множество различных возможных состояний, число которых конечно (хотя и может быть очень большим). Мы назовем их внутренними состояниями устройства. Однако мы не хотим, чтобы объем выполняемых на этом устройстве вычислений был принципиально ограничен. Вспомним описанный выше алгоритм Евклида. В принципе, не существует предельной величины числа, после которой алгоритм перестает работать. Этот алгоритм, или некая общая вычислительная процедура, будет тем же самым независимо от того, сколь велики числа, к которым он применяется. Естественно, для очень больших чисел выполнение процедуры может занять много времени и может потребоваться огромное количество «черновиков» для выполнения пошаговых вычислений. Но сам по себе алгоритм останется тем же конечным набором инструкций, сколь бы большими ни были эти числа.

Значит, несмотря на конечность числа внутренних состояний, наше устройство должно быть приспособлено для работы с входными данными неограниченного объема. Более того, устройство должно иметь возможность использовать внешнюю память неограниченного объема (наши «черновики») для хранения данных, необходимых для вычислений, а также уметь выдавать окончательное решение любого размера. Поскольку наше устройство имеет только конечное число различных внутренних состояний, мы не можем ожидать, что оно будет «хранить внутри себя» все внешние данные, равно как и результаты своих промежуточных вычислений. Напротив, оно должно обращаться только к тем данным и полученным результатам, с которыми оно работает непосредственно в настоящий момент, и уметь производить над ними требуемые (опять же, в данный момент) операции. Далее, устройство записывает результаты этих операций – возможно, в отведенной для этого внешней памяти – и переходит к следующему шагу. Именно неограниченные объемы входных данных, вычислений и окончательного результата говорят о том, что мы имеем дело с идеализированным математическим объектом, который не может быть реализован на практике (рис. 2.3). Но подобная идеализация является очень важной. Чудеса современных компьютерных технологий позволяют создавать электронные устройства хранения информации, которые мы можем рассматривать как неограниченные в приложении к большинству практических задач.

На самом деле память устройства, которая выше была названа «внешней», можно рассматривать как внутренний компонент современного компьютера. Но это уже технические детали – рассматривать часть объема для хранения информации как внутреннюю или внешнюю по отношению к устройству. Одним из способов проводить такое деление между «устройством» и «внешней» частью могло бы стать использование понятий аппаратного (*hardware*) и программного (*software*) обеспечения вычислений. В этой терминологии внутренняя часть могла бы соответствовать аппаратному обеспечению (*hardware*), тогда как внешняя – программному обеспечению (*software*). Я не буду жестко придерживаться именно этой классификации, однако, какую бы точку зрения мы не заняли, не вызывает сомнений, что идеализация Тьюринга достаточно точно аппроксимируется современными электронными компьютерами.

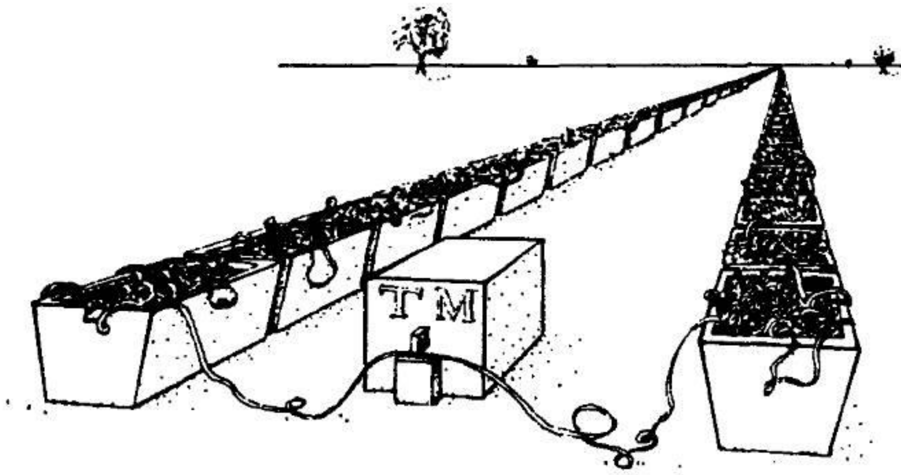


Рис. 2.3. Точная машина Тьюринга требует бесконечной ленты!

Тьюринг представлял внешние данные и объем для хранения информации в виде «ленты» с нанесенными на нее метками. Устройство по мере необходимости могло обращаться к этой ленте, «считывать» с нее информацию и перемещать ее вперед или назад в ходе выполнения операций. Помимо этого, устройство могло ставить новые метки на ленту и стирать с нее старые, что позволяло использовать одну и ту же ленту и как внешнюю память (то есть «черновик»), и как источник входных данных. На самом деле, не стоило бы проводить явное различие между этими двумя понятиями, поскольку во многих операциях промежуточные результаты вычислений могут играть роль новых исходных данных. Вспомним, что при использовании алгоритма Евклида мы раз за разом замещали исходные числа (A и B) результатами, полученными на разных этапах вычислений. Сходным образом та же самая лента может быть использована и для вывода окончательного результата («ответа»). Лента будет двигаться через устройство туда–сюда до тех пор, пока выполняются вычисления. Когда, наконец, все вычисления закончены, устройство останавливается, и результат вычислений отображается на части ленты, лежащей по одну сторону от устройства. Для определенности будем считать, что ответ всегда записывается на части ленты, расположенной слева от устройства, а все исходные числовые данные и условия задачи – на части ленты, расположенной справа от него.

Меня всегда несколько смущало представление о конечном устройстве, которое двигает потенциально бесконечную ленту вперед и назад. Неважно, насколько легок материал ленты – сдвинуть бесконечную ленту все-таки будет трудно! Вместо этого я предпочитаю представлять себе эту ленту как некое окружение, по которому может перемещаться наше конечное устройство. (Конечно же, в современных электронных устройствах ни «лента», ни само «устройство» не должны в обычном смысле физически «перемещаться», но представление о таком «движении» позволяет достичь известной наглядности.) При таком подходе устройство получает все входные данные из этого окружения, использует его в качестве «черновика» и, наконец, записывает в него конечный результат.

В представлении Тьюринга «лента» состоит из бесконечной в обоих направлениях линейной последовательности квадратов. Каждый квадрат либо пуст, либо помечен.[118] Использование помеченных и пустых квадратов означает, что мы допускаем разбиение нашего «окружения» (т.е. ленты) на части и возможность его описания множеством дискретных элементов (в противоположность непрерывному описанию). Это представляется вполне разумным, если мы хотим, чтобы наше устройство работало надежно и совершенно определенным образом. В силу используемой математической идеализации мы допускаем (потенциальную)

[118] На самом деле Тьюринг использовал более сложные записи на ленте, но это не имеет принципиального значения, поскольку такие записи всегда могут быть представлены в виде последовательностей меток (одного типа) и пробелов. Я и далее, когда это допустимо, буду довольно свободно обращаться с исходными определениями Тьюринга.

бесконечность «окружения», однако в каждом конкретном случае входные данные, промежуточные вычисления и окончательный результат всегда должны быть конечными. Таким образом, хотя лента и имеет бесконечную длину, на ней должно быть конечное число непустых квадратов. Другими словами, и с той, и с другой стороны от устройства найдутся квадратики, после которых лента будет абсолютно пустой. Мы обозначим пустые квадраты символом «0», а помеченные – символом «1», например:

..000111101001110010010110100..

Нам нужно, чтобы устройство «считывало» информацию с ленты. Мы будем считать, что оно считывает по одному квадрату за раз и смещается после этого ровно на один квадрат влево или вправо. При этом мы не утрачиваем общности рассуждений: устройство, которое читает за один раз *n* квадратов или перемещается на *k* квадратов, легко моделируется устройством, указанным выше. Передвижение на *k* квадратов можно построить из *k* перемещений по одному квадрату, а считывание *n* квадратов за один прием сводится к запоминанию результатов *n* однократных считываний.

Что именно может делать такое устройство? Каким образом в самом общем случае могло бы функционировать устройство, названное нами «механическим»? Вспомним, что число внутренних состояний нашего устройства должно быть конечным. Всё, что нам надо иметь в виду помимо этого – это то, что поведение нашего устройства полностью определяется его внутренним состоянием и входными данными. Входные данные мы упростили до двух символов – «0» и «1». При заданном начальном состоянии и таких входных данных устройство должно работать совершенно определенным образом: оно переходит в новое состояние (или остается в прежнем), заменяет считанный символ 0 или 1 тем же или другим символом 1 или 0, передвигается на один квадрат вправо или влево, и наконец, оно решает, продолжить вычисления или же закончить их и остановиться.

Чтобы явно определить операции, производимые нашим устройством, для начала пронумеруем его внутренние состояния, например: 0, 1, 2, 3, 4, 5. Тогда действия нашего устройства, или машины Тьюринга, полностью определялись бы неким явным списком замен, например:

0**0** → 0**0**R
0**1** → 13**1**L
1**0** → 65**1**R
1**1** → 1**0**R
2**0** → 0**1**R.STOP
2**1** → 66**1**L
3**0** → 37**0**R
. .
. .
. .
210**0** → 3**1**L
. .
. .
. .
258**1** → 0**0**R.STOP
259**0** → 97**1**R
259**1** → 0**0**R.STOP

Выделенная цифра слева от стрелки – это символ на ленте, который устройство в данный момент считывает. Оно заменяет этот символ выделенной цифрой в середине справа от стрелки. R означает, что устройство должно переместиться вдоль ленты на один квадрат вправо, а L соответствует такому же перемещению влево. (Если, в соответствии с исходным представлением Тьюринга, мы полагаем, что движется не устройство, а лента, то R означает перемещение ленты на один квадрат влево, а L – вправо.) Слово STOP означает, что вычисления завершены и устройство должно остановиться. Например, вторая инструкция 01 → 131L говорит о том, что если устройство находится в начальном состоянии 0 и считывает с ленты 1, то оно должно перейти в состояние 13, оставить на ленте тот же символ **1** и переместиться по ленте на один квадрат влево. Последняя же инструкция 259**1** → 0**0**R.STOP говорит о том, что если устройство находится в состоянии 259 и считывает с ленты **1**, то оно должно вернуться в состояние 0, стереть с ленты **1**, т.е. записать в текущий квадрат **0**, переместиться по ленте на один квадрат вправо и прекратить вычисления.

Вместо номеров 0, 1, 2, 3, 4, 5, ... для обозначения внутренних состояний мы можем – и это более соответствовало бы знаковой системе нанесения меток на ленту – прибегнуть к системе нумерации, построенной только на символах «0» и «1». Состояние *n* можно было бы обозначить просто последовательностью из *n* единиц, но такая запись неэффективна. Вместо этого мы используем двоичную систему счисления, ставшую теперь общепринятой:

$$
\begin{aligned}
0 &\to 0,\\
1 &\to 1,\\
2 &\to 10,\\
3 &\to 11,\\
4 &\to 100,\\
5 &\to 101,\\
6 &\to 110,\\
7 &\to 111,\\
8 &\to 1000,\\
9 &\to 1001,\\
10 &\to 1010,\\
11 &\to 1011,\\
12 &\to 1100 \quad \text{и т. д.}
\end{aligned}
$$

Здесь последняя цифра справа соответствует «единицам» точно так же, как и в стандартной (десятичной) системе записи, но цифра прямо перед ней показывает число «двоек», а не «десятков». В свою очередь третья цифра справа относится не к «сотням», а к «четверкам»; четвертая – к «восьмеркам», а не к «тысячам» и т.д. При этом разрядность каждой последующей цифры (по мере продвижения влево) дается соответственной степенью двойки: 1, 2, 4 (= 2 × 2), 8 (= 2 × 2 × 2), 16 (= 2 × 2 × 2 × 2), 32 (= 2 × 2 × 2 × 2 × 2). (В дальнейшем нам будет иногда удобно использовать в качестве основания системы счисления числа, отличные от «2» и «10». Например, запись десятичного числа 64 по основанию «*три*» даст 2101, где каждая цифра теперь – некоторая степень тройки: $64 = (2 \times 3^3) + 3^2 + 1$; см. главу 4, сноску на с. 96.)

Используя двоичную запись для внутренних состояний, можно представить вышеприведенную инструкцию, описывающую машину Тьюринга, следующим образом:

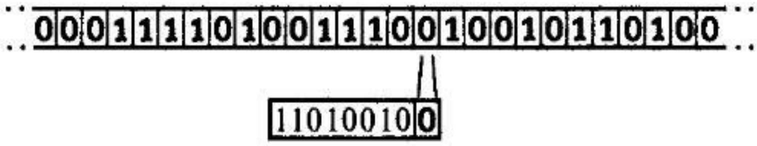
$$
\begin{aligned}
0\mathbf{0} &\to 0\mathbf{0}\mathrm{R}\\
0\mathbf{1} &\to 1101\mathbf{1}\mathrm{L}\\
1\mathbf{0} &\to 1000001\mathbf{1}\mathrm{R}\\
1\mathbf{1} &\to 1\mathbf{0}\mathrm{R}\\
10\mathbf{0} &\to 0\mathbf{1}\mathrm{STOP}\\
10\mathbf{1} &\to 1000010\mathbf{1}\mathrm{L}\\
11\mathbf{0} &\to 100101\mathbf{0}\mathrm{R}\\
&\;\;\vdots\\
11010010\mathbf{0} &\to 11\mathbf{1}\mathrm{L}\\
&\;\;\vdots\\
100000010\mathbf{1} &\to 0\mathbf{0}\mathrm{STOP}\\
100000011\mathbf{0} &\to 1100001\mathbf{1}\mathrm{R}\\
100000011\mathbf{1} &\to 0\mathbf{0}\mathrm{STOP}
\end{aligned}
$$

Здесь я к тому же сократил R.STOP до STOP, поскольку мы вправе считать, что L.STOP никогда не происходит, так как результат последнего шага вычислений, будучи частью окончательного ответа, всегда отображается слева от устройства.

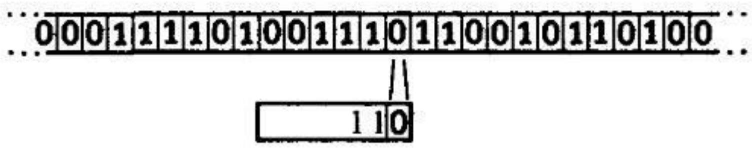
Предположим, что наше устройство находится во внутреннем состоянии, представленном бинарной последовательностью 11010010, и процессу вычисления соответствует участок ленты, изображенный на с. 46. Пусть мы задаем команду

110100100 → 111L.

Та цифра на ленте, которая в данный момент считывается (в нашем случае цифра «0»), показана «жирным» символом справа от последовательности нулей и единиц, обозначающих внутреннее состояние.



В частично описанном выше примере машины Тьюринга (который я выбрал более-менее произвольно) считанный «0» был бы тогда замещен на «1», внутреннее состояние поменялось бы на «11» и устройство переместилось бы на один шаг влево:



Теперь устройство готово к считыванию следующей цифры, снова «0». Согласно таблице, оно оставляет этот «0» нетронутым, но изменяет свое внутреннее состояние на «100101» и передвигается по ленте назад, т.е. на один шаг вправо. Теперь оно считывает «1» и находит где-то ниже в таблице инструкцию, которая определяет изменение внутреннего состояния и указывает, должна ли быть изменена считанная цифра и в каком направлении по ленте должно дальше двигаться устройство. Таким образом устройство будет действовать до тех пор, пока не достигнет команды STOP. В этой точке – после еще одного шага вправо – раздастся звонок, оповещающий оператора о том, что вычисления завершены.

Мы будем считать, что машина всегда начинает с внутреннего состояния «0» и что вся лента справа от устройства изначально пуста. Все инструкции и данные подаются в устройство с правой стороны. Как упоминалось ранее, эта информация всегда имеет форму конечной строки из нулей и единиц, за которой следует пустая лента (т.е. нули). Когда машина получает команду STOP, результаты вычислений оказываются на ленте слева от считывающего устройства.

Поскольку мы хотели бы иметь возможность вводить в устройство и числовые данные, то нам потребуется некий способ описания обычных чисел (под которыми я здесь имею в виду целые неотрицательные числа 0, 1, 2, 3, 4, ...) как части входной информации. Для представления числа *n* можно было бы просто использовать строку из *n* единиц (хотя при этом могут возникнуть трудности, когда речь зайдет о нуле):

1 → **1**,
2 → **11**,
3 → **111**,
4 → **1111**,
5 → **11111** и т. д.

Эта примитивная схема нумерации называется (хотя и довольно нелогично) <u>унарной</u> (<u>единичной</u>) системой. В этом случае символ 0 мог бы использоваться в качестве пробела для разделения двух разных чисел. Наличие такого способа разделения для нас существенно, так как многие алгоритмы оперируют не отдельными числами, а <u>множествами чисел</u>. Например, для выполнения алгоритма Евклида наше устройство должно производить определенные действия над <u>парой</u> чисел A и B. Соответствующая машина Тьюринга может быть легко записана в явном виде. В качестве упражнения заинтересованный читатель может проверить, что нижеследующий набор инструкций действительно описывает машину Тьюринга (которую я буду называть EUC), выполняющую алгоритм Евклида, если в качестве исходных данных использовать два «унарных» числа, разделенных символом **0**:

0**0** → 0**0**R
0**1** → 1**1**L
1**0** → 10**1**R
1**1** → 1**1**L
10**0** → 1010**0**R
10**1** → 11**0**R
11**0** → 100**0**R
11**1** → 11**1**R
100**0** → 100**0**R
100**1** → 101**0**R
101**0** → 111**0**L
101**1** → 110**1**L
110**0** → 110**0**L
110**1** → 1**1**L
111**0** → 111**0**L
111**1** → 1000**1**L
1000**0** → 1001**0**L
1000**1** → 1000**1**L
1001**0** → 10**0**R
1001**1** → 1**1**L
1010**0** → 0**0**STOP
1010**1** → 1010**1**R

Однако я бы порекомендовал такому читателю начать не с этого упражнения, а с чего-нибудь гораздо более простого, например, с машины Тьюринга UN+1, которая просто прибавляет единицу к числу в унарном представлении:

0**0** → 0**0**R
0**1** → 1**1**R
1**0** → 0**1**STOP
1**1** → 1**1**R

Чтобы убедиться в том, что UN+1 на самом деле производит такую операцию, давайте мысленно применим ее, скажем, к ленте вида

... 00000111100000...,

соответствующей числу четыре. Мы будем полагать, что наше устройство сначала находится где-то слева от последовательности единиц. Находясь в исходном состоянии 0, оно считывает **0**, в соответствии с первой инструкцией сохраняет его неизмененным, после чего перемещается на шаг вправо, оставаясь во внутреннем состоянии 0. Оно продолжает последовательно передвигаться вправо до тех пор, пока не встретит первую единицу. После этого вступает в силу вторая инструкция: устройство оставляет единицу как есть и сдвигается на шаг вправо, но уже в состоянии 1. В соответствии с четвертой инструкцией оно сохраняет внутреннее состояние 1, равно как и все считываемые единицы, двигаясь вправо до встречи с первым после набора единиц нулем. Тогда начинает действовать третья инструкция, согласно которой устройство заменяет этот нуль на **1**, перемещается на один шаг вправо (вспомним, что команда STOP эквивалентна R.STOP) и останавливается. Тем самым к последовательности из четырех единиц прибавляется еще одна, превращая – как и требовалось – 4 в 5.

В качестве несколько более трудного упражнения можно проверить, что машина UN×2, определяемая набором инструкций

00 → 00R
01 → 10R
10 → 101L
11 → 11R
100 → 110R
101 → 1000R
110 → 01STOP
111 → 111R
1000 → 1011L
1001 → 1001R
1010 → 101L
1011 → 1011L

<u>удваивает</u> унарное число, как и должно быть, судя по ее названию.

Чтобы понять, как работает машина EUC, нужно явным образом задать пару подходящих чисел, скажем, 6 и 8. Как и ранее, изначально машина находится во внутреннем состоянии 0 и расположена слева, а лента выглядит следующим образом:

... 0000011111101111111100000....

После того, как машина Тьюринга после большого числа шагов останавливается, мы получаем ленту с записью вида

... 000011000000000000...,

при этом машина располагается справа от ненулевых цифр. Таким образом, найденный наибольший общий делитель равен 2 (как и должно быть).

Исчерпывающее объяснение, почему машина EUC (или UN×2) на самом деле осуществляет действие, для которого она предназначена, включает в себя некоторые тонкости, и разобраться в нем, может быть, даже труднее, чем понять устройство самой машины – довольно обычная ситуация с компьютерными программами! (Чтобы полностью понять, почему алгоритмические процедуры делают то, что от них ожидается, необходима определенная интуиция. А не являются ли интуитивные прозрения сами алгоритмическими? Это один из вопросов, которые будут для нас важны в дальнейшем.) Я не буду пытаться дать здесь такое объяснение для приведенных примеров EUC или UN×2. Читатель, шаг за шагом проверив их действие, обнаружит, что я незначительно изменил обычный алгоритм Евклида, чтобы получить более компактную запись в рамках используемой схемы. И всё же описание EUC остается достаточно сложным, включая в себя 22 элементарные инструкции для 11 различных внутренних состояний. В основном эти сложности носят чисто организационный характер. Можно отметить, например, что из этих 22 инструкций только 3 в действительности изменяют запись на ленте! (Даже для UN×2 я использовал 12 инструкций, половина из которых меняют запись на ленте.)

§2.3. Двоичная запись цифровых данных

Унарная система чрезвычайно неэффективна для записи больших чисел. Поэтому мы по большей части будем использовать вышеописанную <u>двоичную</u> систему. Однако, сделать это напрямую и попытаться читать ленту просто как двоичное число мы не сможем. Дело в том, что мы не имеем возможности сказать, когда кончается двоичное представление числа и начинается бесконечная последовательность нулей справа, которая отвечает пустой ленте. Нам нужен способ как-то обозначать конец двоичной записи числа. Более того, часто нам будет нужно вводить в машину <u>несколько</u> чисел, как, например, в случае с алгоритмом Евклида, когда требуется <u>пара</u> чисел.[119] Но в двоичном представлении мы не можем отличить <u>пробелы</u> между числами от нулей или строчек нулей, входящих в записи этих двоичных чисел. К тому же, помимо чисел нам может понадобиться и запись всевозможных сложных инструкций на той же ленте. Для того, чтобы преодолеть эти трудности, воспользуемся процедурой, которую я буду в дальнейшем называть

---

[119] Существует немало других известных в математике способов записи пар, троек и большего количества чисел в виде одного числа, но они менее удобны для наших целей. Например, формула $(1/2)((a + b^2) + 3a + b)$ однозначно представляет пару (a, b) как одно натуральное число. Проверьте сами!

сокращением и согласно которой любая строчка нулей и единиц (с конечным числом единиц) не просто считывается как двоичное число, но замещается строкой из нулей, единиц, двоек, троек и т.д. таким образом, чтобы каждое число в получившейся строчке соответствовало числу единиц между соседними нулями в исходной записи двоичного числа. Например, последовательность

010001011010101101000111010101111100110

превратится в

```
010 0 0101101010110100 0 01110101011110 0110
|  | |  |  |   |  |  |  |  |  |  |   |  |  |    |  |  |
1  0 0  1  2   1  1  2  1  0  0  3   1  1  4    0  2
```

Мы теперь можем считывать числа 2, 3, 4, ... как метки или инструкции определенного рода. Действительно, пусть 2 будет просто «запятой», указывающей на пробел между двумя числами, а числа 3, 4, 5, ... могли бы по нашему желанию символизировать различные инструкции или необходимые обозначения, как, например, «минус», «плюс», «умножить», «перейти в позицию со следующим числом», «повторить предыдущую операцию следующее число раз», и т.п. Теперь у нас есть разнообразные последовательности нулей и единиц, разделенные цифрами большей величины. Эти последовательности нулей и единиц будут представлять собой обычные числа, записанные в двоичной форме. Тогда записанная выше строка (при замене двоек «запятыми») примет вид:

(двоичное число 1001) запятая (двоичное число 11) запятая...

Используя обычные арабские числа «9», «3», «4», «0» для записи соответствующих двоичных чисел 1001, 11, 100 и 0, получаем новую запись всей последовательности в виде:

9, 3, 4 (инструкция 3) 3 (инструкция 4) 0.

Такая процедура дает нам, в частности, возможность указывать, где заканчивается запись числа (и тем самым отделять ее от бесконечной полосы пустой ленты справа), просто используя запятую в конце этой записи. Более того, она позволяет закодировать любую последовательность натуральных чисел, записанных в двоичной системе, как простую последовательность нулей и единиц, в которой для разделения чисел мы используем запятые. Посмотрим, как это сделать, на конкретном примере. Возьмем последовательность

5, 13, 0, 1, 1, 4.

В двоичном представлении она эквивалентна последовательности

101, 1101, 0, 1, 1, 100,

что на ленте можно записать с помощью операции расширения (обратной по отношению к описанной выше процедуре сокращения) как

... 0000100101101010010110011010110101101000110000... .

Такое кодирование легко выполнить, если в исходной двоичной записи чисел провести следующие замены:

$$0 \rightarrow 0$$
$$1 \rightarrow 10$$
$$, \rightarrow 110$$

и после этого добавить бесконечные последовательности нулей с обеих сторон вновь полученной записи. Чтобы сделать более понятной эту процедуру в применении к нашему примеру, разделим полученные двоичные числа пробелами:

... 0000 10 0 10 110 10 10 0 10 110 0 110 10 110 10 110 10 0 0 110 00... .

Я буду называть этот способ представления (наборов) чисел расширенной двоичной записью. (Так, в частности, в расширенной двоичной форме записи число 13 выглядит как 1010010.)

Есть еще одно, последнее, замечание, которое надо сделать в связи с этой системой записи. Это не более, чем техническая деталь, но она необходима для полноты изложения.[120] Двоичная

---

[120] В изложенном выше я не вводил никакой метки для начала последовательности чисел (или инструкций и т.п.). Это совершенно не требуется для входных данных, поскольку всё начинается в тот момент, когда считана первая единица. Однако для конечного результата может понадобиться что-то дополнительное, поскольку *априори* никто не может сказать, как долго придется двигаться по ленте, чтобы

(или десятичная) запись натуральных чисел в некоторой степени избыточна в том смысле, что нули, расположенные слева от записи числа, «не считаются» и обычно опускаются, так что 00110010 представляет собой то же самое двоичное число, что и 110010 (а 0050 – то же самое десятичное число, что и 50). Эта избыточность распространяется и на нуль, который может быть записан и как 000, и как 00, и, конечно, как 0. На самом деле и пустое поле, если рассуждать логически, должно обозначать нуль! В обычном представлении это привело бы к большой путанице, но в описанной выше системе кодирования никаких затруднений не возникает: нуль между двумя запятыми можно записать просто в виде двух запятых, следующих подряд („). На ленте такой записи будет соответствовать код, состоящий из двух пар единиц, разделенных одним нулем:

… 001101100… .

Тогда исходный набор из шести чисел может быть записан в двоичной форме как

101,1101„1,1,100,

и на ленте при кодировании в расширенной двоичной форме мы получим последовательность

… 00001001011010100101101101011010110100011000…,

в которой на один нуль меньше по сравнению с предыдущим кодом того же набора.

Теперь мы можем рассмотреть машину Тьюринга, реализующую, скажем, алгоритм Евклида в применении к паре чисел, записанных в расширенной бинарной форме. Для примера возьмем ту же пару чисел – 6 и 8, которую мы брали ранее. Вместо прежней унарной записи

… 00000111111011111111100000…

воспользуемся двоичным представлением 6 и 8, т.е. 110 и 1000, соответственно. Тогда эта пара имеет вид 6, 8, или в двоичной форме 110, 1000, и в расширенной двоичной записи на ленте она будет выглядеть следующим образом

… 00000101001101000011000000…

Для этой конкретной пары чисел двоичная форма записи не дает никакого выигрыша по сравнению с унарной. Предположим, однако, что мы берем для вычислений (десятичные) числа 1583169 и 8610. В двоичной записи они имеют вид

110000010100001000001, 10000110100010.

На ленте при расширенном двоичном кодировании им будет соответствовать последовательность

… 001010000000100100000010000000101101000000101001000000100110

которая занимает менее двух строк, тогда как для унарной записи пары чисел «1583169, 8610» не хватило бы места на страницах этой книги!

Машину Тьюринга, выполняющую алгоритм Евклида для чисел, записанных в расширенной двоичной форме, при желании можно получить из EUC с помощью пары дополнительных алгоритмов, которые переводили бы числа из расширенной двоичной формы в унарную и обратно. Однако, такой подход чрезвычайно неэффективен, ибо громоздкость унарной системы записи была бы по-прежнему «внутренне» присуща всему устройству, что проявилось бы в его низком быстродействии и потребности в огромном количестве «черновиков» (на левой стороне ленты). Можно построить и более эффективную машину Тьюринга для алгоритма Евклида, оперирующую исключительно расширенными двоичными числами, но для понимания принципов ее работы это не особенно важно.

Для того, чтобы показать, каким образом машина Тьюринга может работать с числами в расширенном двоичном представлении, обратимся к значительно более простой, чем алгоритм Евклида, процедуре – просто прибавлению единицы к произвольному натуральному числу. Ее можно выполнить с помощью следующей машины Тьюринга (которую я назову XN+1):

добраться до первой (т.е. самой левой!) единицы. Хотя при движении налево может встретиться длинная строка нулей, нет никаких гарантий, что еще <u>дальше</u> не встретится единица. В этом случае применимы различные подходы. Можно было бы всегда использовать специальную отметку (допустим, 6, записанную при помощи процедуры «сокращения»), чтобы указывать начало и завершение окончательного ответа. Но для простоты я в своем изложении буду придерживаться другой точки зрения, согласно которой мы всегда «знаем», сколько в действительности ленты обработало наше устройство (например, можно представить, что оно оставляет своего рода «след»), так что не обязательно просматривать ленту до бесконечности, чтобы убедиться в том, что весь ответ считан.

00 → 00R
01 → 11R
10 → 00R
11 → 101R
100 → 110L
101 → 101R
110 → 01STOP
111 → 1000L
1000 → 1011L
1001 → 1001L
1010 → 1100R
1011 → 101R
1101 → 1111R
1110 → 111R
1111 → 1110R

И вновь некоторые дотошные читатели могут захотеть проверить, вправду ли эта машина Тьюринга действует так, как должна, если взять, скажем, число 167. Это число имеет двоичное представление 10100111 и записывается на ленте как

... 0000100100010101011000... .

Чтобы прибавить единицу к двоичному числу, мы просто находим в его записи последний нуль и меняем его на единицу, а все непосредственно следующие за ним единицы – на нули. Так что 167 + 1 = 168 в двоичной форме записывается в виде

10100111 + 1 = 10101000.

Таким образом, наша «прибавляющая единицу» машина Тьюринга должна превратить предыдущую запись на ленте в

... 0000100100100001100000

что она и делает.

Обратите внимание, что даже самая простая операция прибавления единицы в такой записи выглядит довольно сложно, включая в себя 15 инструкций и восемь различных внутренних состояний! Конечно, в случае унарной записи всё было значительно проще, поскольку тогда «прибавление единицы» означало удлинение строчки единиц еще на одну, поэтому не удивительно, что машина UN+1 была более простой. Однако, для очень больших чисел UN+1 была бы слишком медленной из-за чрезмерной длины ленты, и тогда более сложная машина XN+1, но работающая с более компактным расширенным двоичным представлением, оказалась бы предпочтительнее.

Несколько отступая в сторону, я укажу операцию, для которой машина Тьюринга проще в расширенной двоичной, нежели в унарной форме – это <u>умножение на два</u>. Действительно, машина Тьюринга XN×2, заданная в виде

00 → 00R
01 → 10R
10 → 01R
11 → 100R
100 → 111R
110 → 01STOP

запросто выполнит эту операцию в расширенной двоичной форме, тогда как соответствующая унарная машина UN×2, описанная ранее, гораздо сложнее!

Этот раздел дает определенное представление о том, на что способны в простейших случаях машины Тьюринга. Как и следовало ожидать, при выполнении более или менее сложных операций эти машины могут становиться, и действительно становятся, несравненно более сложными. Каковы же принципиальные возможности таких устройств? Мы рассмотрим этот вопрос в следующем параграфе.

§2.4. Тезис Черча–Тьюринга

После ознакомления с принципами построения простых машин Тьюринга легко убедиться, что все основные математические операции, такие как сложение двух чисел, их перемножение или возведение одного из них в степень другого, могут на самом деле быть выполнены соответствующими машинами Тьюринга. Построение таких машин в явном виде не представляет больших затруднений, но я не собираюсь сейчас этим заниматься. Машины Тьюринга могут выполнять операции, результат которых выражается парой натуральных чисел, например, деление с остатком, или сколь угодно большим, но конечным множеством чисел. Более того, можно сконструировать такие машины Тьюринга, для которых арифметические операции не предопределены заранее, а могут задаваться инструкциями, вводимыми с ленты. При этом возможно, что та конкретная операция, которая должна быть выполнена, будет зависеть в тот или иной момент от результатов вычислений, которые машина должна была выполнить на предыдущих этапах. («Если результат вычислений больше, чем то-то, надо сделать то-то, в противном случае выполнить то-то».) Убедившись, что можно построить машины Тьюринга, выполняющие арифметические или простые логические операции, уже не так трудно представить себе, какими должны быть машины, выполняющие более сложные задачи алгоритмического характера. «Повозившись» немного с подобными задачами, легко приходишь к убеждению в том, что машина этого типа может выполнять вообще любые механические операции! Тогда с точки зрений математики приобретает смысл определение механической операции как такой операции, которую может выполнить подобная машина. Существительное «алгоритм» и прилагательные «вычислимый», «рекурсивный» и «эффективный» используются математиками для обозначения механических операций, которые могут быть выполнены теоретическими устройствами такого рода, т.е. машинами Тьюринга. Если некоторая процедура четко определена и по природе своей механистична, то можно вполне обоснованно предположить, что найдется машина Тьюринга, способная ее выполнить. Это, в конце концов, и есть основной момент наших (то есть Тьюринга) рассуждений, лежащий и в основе самой концепции машины Тьюринга.

С другой стороны, остается ощущение, что принципы построения этих машин содержат излишние ограничения. Разрешение устройству считывать за один раз только одну двоичную цифру (0 или 1) и передвигаться каждый раз только на один шаг, да еще вдоль единственной одномерной ленты, на первый взгляд, ограничивает возможности машины. Почему бы не разрешить одновременное использование четырех, пяти или, возможно, тысячи разных лент, по которым одновременно двигалось бы большое количество взаимосвязанных считывающих устройств? Почему бы не ввести целую плоскость с нулями и единицами (или, например, трехмерное пространство), вместо того, чтобы настаивать на использовании одномерной ленты? Почему бы не использовать другие системы счисления или символы из каких-нибудь более сложных алфавитов? По сути, ни одно из этих изменений ни в малейшей степени не влияет на то, что в принципе может быть достигнуто с помощью машины Тьюринга, хотя некоторые из них отразились бы на экономичности производимых операций (как это наверняка произошло бы, разреши мы использование нескольких лент). Класс осуществляемых операций, попадающих, таким образом, под определение «алгоритма» (или «вычисления», или «выполнимой процедуры», или «рекурсивной операции»), остался бы в точности тем же самым, если мы расширим определение наших машин и включим в него даже все предлагавшиеся выше модификации одновременно!

Мы можем видеть, что нет необходимости в дополнительных лентах, коль скоро устройство может по мере надобности находить свободное место на одной ленте. При этом может потребоваться постоянная перезапись данных с одного места ленты на другое. Это, может быть, «неэффективно», но в принципе не ограничивает возможности машин Тьюринга.[121] Сходным образом, использование более чем одного устройства Тьюринга для параллельных вычислений – идея, ставшая очень популярной в последние годы в связи с попытками более

[121] Один из способов записи информации с двух лент на одну – вставить записи одной из них между записями другой. При этом нечетные отметки на новой ленте могут соответствовать отметкам первой ленты, тогда как четные – отметкам второй. Аналогичная схема работает и для четырех, и для большего числа лент. «Неэффективность» этой процедуры обусловлена тем, что считывающему устройству пришлось бы «прыгать» взад–вперед по ленте, оставляя на ней маркеры как на четных местах, так и на нечетных, с тем, чтобы фиксировать свое положение в каждый момент.

точного моделирования человеческого мозга, – не дает никаких принципиальных преимуществ (хотя при определенных обстоятельствах может увеличиться быстродействие). Использование двух непосредственно не связанных друг с другом устройств не даст выигрыша по сравнению с двумя взаимосвязанными устройствами.[122] Но если два устройства связаны друг с другом, то, в сущности, это уже одно устройство!

А что можно сказать об ограничении Тьюринга, касающегося одномерности ленты? Если мы считаем, что эта лента представляет собой «окружение», то, возможно, мы бы предпочли в качестве такового иметь плоскую поверхность, или, допустим, трехмерное пространство. Может показаться, что плоскость лучше подошла бы для изображения «блок-схемы» вычислений (как в вышеприведенном описании последовательности действий алгоритма Евклида), чем одномерная лента.[123] Однако запись блок-схемы в «одномерной» форме не представляет принципиальных трудностей (например, можно использовать обычное словесное описание). Двумерное плоское изображение дает только удобство и простоту восприятия, но, по сути, ничего не меняет. Всегда есть возможность преобразовать координаты отметки или объекта на двумерной плоскости или в трехмерном пространстве и явным образом отобразить их на одномерной ленте. (Фактически, использование двумерной плоскости полностью эквивалентно использованию двух лент. Две ленты дают две «координаты», которые нужны для определения местоположения точки на двумерной плоскости; аналогично, три ленты могут выполнять ту же роль для точки в трехмерном пространстве.) И хотя эта одномерная запись может вновь оказаться «неэффективной», принципиальные возможности устройства это никак не ограничивает.

Несмотря на всё это, по-прежнему остается вопрос о том, действительно ли понятие машины Тьюринга охватывает все логические или математические операции, которые мы могли бы назвать «механическими». В то время, когда Тьюринг написал свою основополагающую работу, ситуация была гораздо менее ясной, чем сегодня, поэтому Тьюринг справедливо посчитал необходимым предоставить развернутое изложение этого вопроса. Детально рассмотренная Тьюрингом проблема получила дополнительное обоснование благодаря тому, что совершенно независимо от Тьюринга (и на самом деле несколько ранее) американский логик Алонзо Черч (совместно со Стивеном Клини), стремясь найти решение проблемы алгоритмической разрешимости Гильберта, предложил свою схему лямбда-исчисления. Хотя то, что это была всеобъемлющая полностью механическая схема, было не так очевидно, как в случае с подходом Тьюринга, ее несомненным преимуществом была удивительная компактность математической структуры. (Я буду рассматривать замечательный анализ Черча в конце главы.) Независимо от Тьюринга были предложены и другие подходы к решению задачи Гильберта (см.

---

[122] **В.Э.:** Не даст выигрыша КОМУ и В ЧЁМ?... Вообще в этих словах Пенроуза отражается его способ мышления и те ограничения, которые он сам на себя накладывает. По этим словам видно, что ход его мыслей на самом деле таков: «Вот, есть какой-то алгоритм (типа алгоритма Евклида или т.п.); нам этот алгоритм нужно выполнить на каком-то «механическом устройстве»»; вот, тогда действительно выполнение этого (одного!) алгоритма на двух машинах не даст никакого «выигрыша» по сравнению с выполнением его на одной машине. Сначала есть алгоритм – и он один... Но такой установкой и такими рамками рассуждений Пенроуз отгораживает себя от той сущности, для чего вообще (Природе и людям) нужны параллельные процессоры. Они в первую очередь нужны для того, чтобы выполнять разную работу. Например, процессор, встроенный в глазу человека (и других животных) уже сам (еще до мозга) выполняет большую работу по обеспечению зрения: ловит изменения освещения (то есть, движения объектов) и т.п., а в мозг (другой, параллельный процессор) поступает уже частично обработанная информация (которую мозг обрабатывает дальше – опять же многими параллельными процессорами). Эти параллельные компьютеры делают каждый свою работу и сигнализируют друг другу о результатах. Для одного компьютера сигналы другого компьютера поступают как ВНЕШНЯЯ информация (и асинхронная – т.е. никак не вызванная прежней работой этого компьютера). Всё это взаимодействие множества таких асинхронно работающих компьютеров невозможно ни отобразить, ни понять при размышлениях и рассуждениях в рамках ОДНОЙ «машины Тьюринга». А когда ты не понял, как эта система параллельных компьютеров работает, тогда и приходишь к убеждению – как Пенроуз, – что природу разума ну никак невозможно «объяснить алгоритмически».

[123] В согласии с предложенным здесь описанием, эта блок-схема была бы скорее частью «устройства», нежели внешнего окружения – «ленты». На ленте мы до сих пор отображали только числа A, B, A–B, и т.п. Однако в дальнейшем нам потребуется также возможность описания и самого устройства в линейной одномерной форме. Как мы увидим далее в связи с универсальной машиной Тьюринга, есть тесная взаимосвязь между свойствами конкретного «устройства» и свойствами возможных «данных» (или «программы») для него. Поэтому удобно в обоих случаях придерживаться одномерной формы записи.

Ганди [1988]), среди которых можно выделить работу американского логика польского происхождения Эмиля Поста (опубликованную несколько позже работы Тьюринга, но содержащую идеи, более близкие идеям Тьюринга, нежели Черча). В скором времени было доказано, что все эти схемы совершенно эквивалентны.

Это значительно укрепило точку зрения, известную как тезис Черча–Тьюринга, которая утверждает, что машина Тьюринга (или ее эквивалент) на самом деле определяет то, что в математике понимают под алгоритмической (или выполнимой, или рекурсивной, или механической) процедурой. Сегодня, когда быстродействующие электронные компьютеры прочно вошли в нашу жизнь, немного найдется тех, кто считает необходимым ставить под сомнение эту теорию в ее изначальной формулировке. Вместо этого сейчас исследователи обратили внимание на вопрос, какие логические и математические операции могут выполнять реальные физические системы (возможно, включающие и человеческий мозг), подчиняющиеся точным физическим законам: точно такие же, что и машины Тьюринга, или же их возможности больше или меньше? Что касается меня, то я с удовольствием принимаю исходную математическую интерпретацию тезиса Черча–Тьюринга. С другой стороны, вопрос о его отношении к поведению реальных физических систем заслуживает отдельного рассмотрения и будет занимать в дальнейшем центральное место в наших рассуждениях.

§2.5. Числа, отличные от натуральных

В предыдущих параграфах мы рассматривали действия над натуральными числами и отметили тот замечательный факт, что машина Тьюринга может оперировать с натуральными числами произвольной величины, несмотря на то, что каждая машина имеет фиксированное и конечное число внутренних состояний. Однако часто возникает необходимость в операциях с более сложными числами, такими как отрицательные числа, обыкновенные дроби и бесконечные десятичные дроби. Первые две категории (т.е. числа вида –597/26) легко поддаются обработке машинами Тьюринга, причем и числители, и знаменатели могут быть сколь угодно большими. Всё, что для этого нужно – какой-нибудь подходящий код для знаков «–» и «/», который можно легко выбрать при использовании расширенной двоичной записи (например, «3» = `1110` для знака «–», а «4» = `11110` – для знака «/»). Таким образом, отрицательные числа и обыкновенные дроби рассматриваются как конечные наборы натуральных чисел, и с точки зрения общих вопросов вычислимости ничего нового не дают.

То же можно сказать и о конечных десятичных выражениях с произвольным числом знаков после запятой, поскольку они представляют собой лишь частный случай обыкновенных дробей. Так, например, конечная десятичная аппроксимация иррационального числа π, заданная числом 3,14159265, есть просто дробь 314159265/100000000. Однако бесконечные десятичные выражения, такие как полная запись числа π

$$\pi = 3{,}14159265358979...,$$

представляют определенные трудности. На самом деле, ни входные, ни выходные данные машины Тьюринга не могут быть бесконечными десятичными выражениями. Можно было бы думать, что нашлась бы машина Тьюринга, способная выдавать одну за другой все последовательные цифры – 3, 1, 4, 1, 5, 9, ... в десятичной записи числа π и переносить их на выходную ленту, а мы просто позволим этой машине работать бесконечно долго. Но это запрещено для машин Тьюринга. Мы должны дождаться остановки машины (сопровождаемой звонком колокольчика!), прежде чем сможем ознакомиться с результатом. До того момента, пока машина не выполнит команды STOP, выходные данные могут изменяться и поэтому не являются достоверными. С другой стороны, после полной остановки машины результат должен быть с необходимостью конечным.

Существует, однако, «законная» процедура для того, чтобы заставить машину Тьюринга последовательно воспроизводить цифры примерно так, как это предлагалось выше. Если мы хотим получить бесконечную десятичную запись, скажем, числа π, мы могли бы заставить машину Тьюринга сначала рассчитать его целую часть, 3, используя на входе 0, затем – первую цифру дробной части, 1, используя на входе 1, затем – вторую цифру дробной части, 4, используя на входе 2, потом – третью цифру, 1, используя 3 и т.д. Вообще говоря, машина Тьюринга для получения всех цифр десятичной записи числа π в этом смысле действительно существует, хотя реализовать ее в явном виде было бы затруднительно. Подобное же замечание относится и ко многим другим иррациональным числам, таким, например, как $\sqrt{2} = 1{,}414213562...$. Однако оказывается – и мы увидим это в следующей главе, – что некоторые иррациональные числа принципиально не могут быть получены с помощью машины Тьюринга. Числа, которые можно получить таким образом, называются вычислимыми (Тьюринг [1937]), а остальные (в действительности абсолютное большинство!)[124] – невычислимыми. Я еще вернусь к этой теме и затрону ряд смежных вопросов в последующих главах. К нам это имеет отношение в связи с вопросом о том, может ли реальный физический объект (например, человеческий мозг) быть адекватно описан в терминах вычислимых математических структур в соответствии с нашими физическими теориями.

Проблема вычислимости важна для математики в целом. Не следует думать, что она относится только к числам как таковым. Ведь машины Тьюринга могут непосредственно оперировать математическими формулами, например, алгебраическими или тригонометрическими выражениями, или выполнять формальные действия математического анализа. Всё, что для этого нужно, это некий способ точного кодирования всех используемых математических символов в виде последовательностей нулей и единиц, которые позволят применить соответствующую машину Тьюринга. Именно это Тьюринг имел в виду, когда он взялся за проблему алгоритмической разрешимости, в которой требуется найти алгоритмическую процедуру для ответа на самые общие математические вопросы. Очень скоро мы вновь обратимся к этой теме.

§2.6. Универсальная машина Тьюринга

Я еще не затрагивал понятия универсальной машины Тьюринга. Лежащий в ее основе принцип понять нетрудно, хотя детали могут быть сложны. Основная идея состоит в том, чтобы закодировать команды для произвольной машины Тьюринга T в виде последовательности нулей и единиц, которую можно записать на ленте. Эта запись используется как начальная часть входных данных для некоторой особой машины Тьюринга U, называемой универсальной, которая затем обрабатывает остальную часть ленты в точности так, как это сделала бы машина T. Универсальная машина Тьюринга – это универсальный имитатор. Начальная часть ленты дает универсальной машине U всю информацию, необходимую для точной имитации любой машины T!

Чтобы показать, как это может быть реализовано, нам потребуется какая-нибудь система нумерации машин Тьюринга.[125] Рассмотрим список инструкций, определяющих произвольную машину Тьюринга, например, одну из описанных выше. Мы должны в соответствии с некоторыми четкими правилами представить эти инструкции в виде последовательностей нулей и единиц. Это можно сделать, например, с помощью процедуры «сокращения», которую мы использовали ранее. Тогда, если мы закодируем символы R, L, STOP, «стрелка» (→) и «запятая», скажем, числами 2, 3, 4, 5 и 6 соответственно, то мы сможем записать их в виде «сокращений» 110, 1110, 11110, 111110 и 1111110. Цифры 0 и 1, кодируемые, соответственно, как 0 и 10, могут быть использованы для записи строк этих символов, входящих в таблицу действий машины Тьюринга. Нам не нужны различные обозначения для «жирных» цифр 0 и 1 и для остальных цифр в таблице, поскольку расположение «жирных» цифр в конце двоичного кода является достаточным отличительным признаком. При этом 1101, например, будет читаться как двоичное число 1101, представляемое на ленте последовательностью 1010010. В частности, 00 будет читаться как 00, что без всякой двусмысленности можно закодировать как 0 или вовсе опустить. Можно существенно сэкономить, если не кодировать «стрелки» и непосредственно предшествующие им символы, а воспользоваться цифровым упорядочением команд, позволя-

---

[124] **В.Э.:** Разумеется, никаких «невычислимых иррациональных чисел» нет, и мы увидим это в комментариях к следующей главе. «Вывод» о их существовании основывается только на той путанице, которую внес в рассуждениях людей Георг Кантор. Существование «невычислимых» объектов можно установить лишь с той же достоверностью, как существование привидений и летающих на метлах ведьм. Всё это не математика, а мистика, говоря словами Леопольда Кронекера {CANTO2.1504 = МОИ № 39}.

[125] **В.Э.:** В комментариях к книге «Тени разума» я говорил о неопределенности множества рассматриваемых Пенроузом процедур C {PENRS1} и удивлялся, почему математики так держатся за машины Тьюринга {МОИ № 17, стр.66}, хотя для программиста это просто уродины. Теперь это понятно: машины Тьюринга нужны математикам потому, что их можно перенумеровать (хоть как-то: в отличие от абстрактных «программ вообще»). А пронумеровав тогда уже проводить «диагональный процесс», чтобы извлекать из него свои бесценные выводы... ☺

ющим определить, какими должны быть эти символы. Правда, для этого надо убедиться в отсутствии «дырок» в получившемся порядке и добавить, где требуется, «немые» команды. (Например, машина Тьюринга XN+1 не имеет команды, соответствующей коду 110**0**, поскольку такая комбинация в ходе ее работы никогда не встречается. Следовательно, мы должны ввести в список команд немую команду, скажем 110**0** → 0**0**R, которая не вызовет каких бы то ни было изменений в работе машины. Сходным образом мы должны добавить немую команду 10**1** → 0**0**R в список команд машины XN×2.) Без таких «немых» команд кодирование последующих команд было бы нарушено. Как можно видеть, на самом деле мы не нуждаемся и в запятой в конце каждой команды, поскольку символов L и R вполне достаточно для отделения команд друг от друга. Поэтому мы просто будем использовать такую систему кодирования:

0 для 0 или **0**,
10 для 1 или **1**,
110 для R,
1110 для L,
11110 для STOP.

В качестве примера выпишем команды для машины Тьюринга XN+1 (с дополнительной немой командой 110**0** → 0**0**R). Опуская стрелки, цифры, непосредственно предшествующие им, и запятые, получим

0**0**R	1**1**R	0**0**R	10**1**R
11**0**L	10**1**R	0**1**STOP	100**0**L
101**1**L	100**1**L	110**0**R	10**1**R
0**0**R	111**1**R	11**1**R	111**0**R.

Мы можем улучшить полученный результат, если опустим все 0**0** и заменим каждые 0**1** просто единицей в соответствии с тем, что говорилось ранее. Тогда мы получим строку символов

R1**1**RR10**1**R11**0**L10**1**R**1**STOP100**0**L101**1**L100**1**L110**0**R10**1**RR111**1**R11**1**R111**0**R,

которая на ленте записывается как последовательность

110101011011010010110101001110100101101011110100001110100101
011101000101110101000110100101101101010101011010101011010101
00110.

Есть еще два способа немного сэкономить. Во-первых, всегда можно удалить код 110 в начале записи (вместе с бесконечным участком пустой ленты, предшествующим этому коду). Он обозначает последовательность 0**0**R, соответствующую начальной команде 0**0** → 0**0**R, которую я до сих пор неявно считал общей для <u>всех</u> машин Тьюринга, поскольку она необходима для того, чтобы устройство, начав работу в произвольной точке слева от начала записи на ленте, могло перемещаться вправо до тех пор, пока не встретит первую непустую клетку. Во-вторых, точно так же всегда можно удалить код 110 (и неявную бесконечную последовательность нулей, которая, по предположению, следует за ним) в конце записи, поскольку этой кодовой последовательностью должно заканчиваться описание любой машины Тьюринга (во всех случаях список команд заканчивается командой R, L или STOP). Получающееся <u>двоичное число</u> – это номер машины Тьюринга, который для XN+1 будет выглядеть так:

1010110110100101101010011101001011010111101000011101001010111010001011101010001101001011011010101010110101010110101010100.[126]

[126] **В.Э.:** Вообще то, что Пенроуз делает с «машинами Тьюринга», можно проделать и с программами реальных компьютеров, если ограничиться какой-нибудь одной определенной машиной, допустим, IBM/360, причем здесь это получается даже проще. Пусть машина имеет побайтовую организацию памяти и все байты линейно пронумерованы (номер байта называется «адресом»). Рассматриваем всю память как единую последовательность битов (поле длиной n байтов дает $8n$ битов). Считаем все эти биты записью <u>одного</u> большого числа. Таким образом, сам двоичный код программы и есть ее номер среди программ данной машины. Если программа, например, занимает ровно 64K (65536 байтов, 524288 битов) и старший

В обычной десятичной записи этот номер равен

450813704461563958982113775643437908.

Иногда машину с номером *n* мы, не вполне точно, будем называть *n*-й машиной Тьюринга и обозначать ее T_n. В этом случае XN+1 становится 450813704461563958982113775643437908-й машиной Тьюринга!

Кажется поразительным факт, что нам надо пробежать так долго вдоль «списка» машин Тьюринга, чтобы найти машину, выполняющую такую тривиальную операцию, как прибавление единицы к натуральному числу (в расширенном двоичном представлении). (Я не думаю, что моя система кодирования была в целом настолько неэффективна, хотя в ней и есть еще возможности для незначительных улучшений.) В действительности, есть машины Тьюринга и с меньшими номерами, которые представляют интерес, например UN+1 с двоичным номером 101011010111101010, который в десятичной записи превращается всего лишь в 177642. Значит, особенно тривиальная машина UN+1, которая просто дописывает 1 единицу в конце последовательности единиц, является 177642-й машиной Тьюринга. Интересно, что «умножение на два» в списке машин Тьюринга попадает где-то между этими двумя машинами, причем и в унарном, и в расширенном двоичном представлении: номер XN×2 равен 10 389 728 107, а номер UN×2 – 1492 923 420 919 872 026 917 547 669.

Наверное, принимая во внимание величины этих номеров, уже не вызовет удивления тот факт, что абсолютное большинство натуральных чисел не соответствует ни одной рабочей машине Тьюринга. Приведем перечень первых тринадцати машин Тьюринга в соответствии с принятой нумерацией:

$$
\begin{array}{llll}
T_0: & 00 \to 00\mathrm{R}, & 01 \to 00\mathrm{R}, & \\
T_1: & 00 \to 00\mathrm{R}, & 01 \to 00\mathrm{L}, & \\
T_2: & 00 \to 00\mathrm{R}, & 01 \to 01\mathrm{R}, & \\
T_3: & 00 \to 00\mathrm{R}, & 01 \to 00\mathrm{STOP}, & \\
T_4: & 00 \to 00\mathrm{R}, & 01 \to 10\mathrm{R}, & \\
T_5: & 00 \to 00\mathrm{R}, & 01 \to 01\mathrm{L}, & \\
T_6: & 00 \to 00\mathrm{R}, & 01 \to 00\mathrm{R}, & 10 \to 00\mathrm{R}, \\
T_7: & 00 \to 00\mathrm{R}, & 01 \to ???, & \\
T_8: & 00 \to 00\mathrm{R}, & 01 \to 100\mathrm{R}, & \\
T_9: & 00 \to 00\mathrm{R}, & 01 \to 10\mathrm{L}, & \\
T_{10}: & 00 \to 00\mathrm{R}, & 01 \to 11\mathrm{R}, & \\
T_{11}: & 00 \to 00\mathrm{R}, & 01 \to 01\mathrm{STOP}, & \\
T_{12}: & 00 \to 00\mathrm{R}, & 01 \to 00\mathrm{R}, & 10 \to 00\mathrm{R}.
\end{array}
$$

Из этих машин T_0 просто перемещается вправо, стирая всё, что ей попадается на пути, никогда не останавливаясь и не меняя направления движения. Машина T_1 выполняет в сущности ту же операцию, но более громоздким путем, отступая на шаг назад каждый раз, когда она стирает очередную единицу на ленте. Так же как и T_0, машина T_2 двигается вправо, никогда не останавливаясь, но относится к ленте более «почтительно», попросту оставляя всю информацию нетронутой. Эти машины не могут использоваться в качестве машин Тьюринга, поскольку никогда не останавливаются. T_3 – первая в этом списке «правильная» машина: она скромно прекращает действие после того, как изменяет первую (самую левую) единицу на нуль. T_4 сталкивается с серьезной проблемой. Найдя первую единицу на ленте, она переходит во внутреннее состояние, которое нигде не описано, и, следовательно, машина не имеет никаких команд для следующего шага. С той же проблемой сталкиваются T_8, T_9 и T_{10}. С T_7 возникают трудности еще более фундаментального характера. Строка нулей и единиц, которой она представляется, включает последовательность из пяти единиц: 110111110. Интерпретации этой последовательности не существует, поэтому T_7 намертво застревает сразу же, как только доходит до первой единицы. (Я буду называть T_7, равно как и любую другую машину T_n, двоичное расширенное представлений которой содержит более четырех единиц, <u>некорректно определенной</u>.) Машины T_5, T_6, и T_{12} испытывают те же трудности, что и T_0, T_1, T_2: они просто никогда не останавливаются. Все эти машины – T_0, T_1, T_2, T_5, T_6, T_7, T_8, T_9, T_{10} и T_{12} – совершенно

бит первого ее байта равен 1, то номер этой программы (при условии, что ведущие нули можно опускать) будет находиться в пределах между 2^{524287} и $2^{524288}–1$ (включительно).

бесполезные устройства! Только T_3 и T_{11} являются функциональными машинами Тьюринга, да и то не слишком интересными. Причем T_{11} даже скромнее, чем T_3: натолкнувшись на первую же единицу, она останавливается и вообще ничего не меняет!

Надо заметить, что наш перечень содержит избыточную информацию. Машина T_{12} идентична T_6, а по действиям обе они аналогичны T_0, поскольку ни T_6, ни T_{12} никогда не переходят во внутреннее состояние 1. Но нам нет нужды волноваться из-за этой избыточности, равно как из-за изобилия неработоспособных (фиктивных) машин Тьюринга в нашем списке. На самом деле, мы могли бы изменить систему кодирования таким образом, чтобы избавиться от большого числа бесполезных устройств и значительно уменьшить избыточность списка машин. Но всё это можно сделать только ценой усложнения нашей примитивной универсальной машины Тьюринга, которая должна расшифровывать вводимую в нее запись и имитировать машину T_n, чей номер она считала. Это было бы оправдано, если бы было можно избавиться от всех бесполезных (и повторяющихся) машин. Но это, как мы увидим чуть позднее, невозможно! Поэтому мы оставим нашу систему кодирования без изменений.

Будет удобно интерпретировать ленту с последовательностью меток на ней, например,

...0001101110010000...,

как двоичное представление некоторого числа. Вспомним, что нули простираются бесконечно в обе стороны, а вот количество единиц конечно. Кроме того, я буду полагать, что их число отлично от нуля (т.е. что в этой последовательности существует хотя бы одна единица). Мы можем тогда считывать конечную строку символов между первой и последней единицами (включительно), которая в предыдущем случае имеет вид

110111001,

как двоичное представление натурального числа (в десятичной форме это 441). Однако такая процедура даст нам только нечетные числа (их двоичное представление оканчивается на 1), тогда как нам нужна возможность представления всех натуральных чисел. Поэтому мы воспользуемся следующим несложным приемом – будем удалять последнюю единицу (которая принимается просто за маркер, обозначающий конец выражения) и считывать оставшуюся часть как двоичное число.[127] Тогда в последнем примере получим двоичное число

11011100,

которое соответствует десятичному числу 220. Эта процедура имеет то преимущество, что нуль также представляется непустой лентой, а именно:

... 0000001000000....

Рассмотрим, как действует машина Тьюринга T_n, на некоторую (конечную) строку нулей и единиц на ленте, которая подается в устройство справа. Удобно рассматривать эту строку как двоичное представление некоторого числа, например m, в соответствии с приведенной выше схемой. Предположим, что после определенного числа шагов машина T_n в конце концов останавливается (т.е. доходит до команды STOP). Строка двоичных цифр, которые машина выписала к этому моменту на левой части ленты, и будет искомым результатом вычислений. Считывая эту последовательность в соответствии с той же схемой так же как двоичное представление некоторого числа, получим новое число, скажем, p. Тогда мы можем записать соотношение, выражающее тот факт, что результатом действия n-й машины Тьюринга T_n, на число m является число p, следующим образом:

$$T_n(m) = p.$$

Взглянем на это соотношение с несколько иной точки зрения. Мы будем считать, что это выражение описывает некоторую специфическую операцию, которая применяется к паре чисел n и m для того, чтобы получить p. (Это означает: для заданных двух чисел m и n мы можем найти значение p, если введем m в n-ю машину Тьюринга.) Эта специфическая операция является полностью алгоритмической. Поэтому она может быть выполнена одной конкретной машиной Тьюринга U; иными словами, U, совершая действие над парой (n, m), дает в результате p. Поскольку машина U должна производить операцию над обоими числами n и m, чтобы получить ответ, выражаемый одним числом p, то нам нужно придумать способ для записи пары (n, m) на одной ленте. С этой целью предположим, что n записывается в стандартной двоичной форме и заканчивается последовательностью 111110. (Вспомним, что двоичный номер всякой коррект-

[127] Эта процедура имеет отношение только к методу, который позволяет интерпретировать запись на ленте как натуральное число. Она не изменяет номера наших конкретных машин Тьюринга, таких как EUC и XN+1.

но определенной машины Тьюринга, – это последовательность символов, состоящая только из сочетаний вида 0, 10, 110, 1110 и 11110, поэтому он нигде не содержит более четырех единиц подряд. Таким образом, если T_n – корректно определенная машина, то появление последовательности 111110 действительно будет означать конец записи номера *n*.) Всё, что следует за ней, должно быть просто записью числа *m* на ленте в соответствии с приведенными выше правилами (т.е. двоичное число *m* и строка 1000... непосредственно за ним). Таким образом, с этой второй частью ленты машина T_n и должна производить предполагаемые действия.

Если в качестве примера мы возьмем $n = 11$ и $m = 6$, то на ленте, вводимой в машину *U*, мы будем иметь последовательность

…000101111111011010000… .

Она образована из следующих составляющих:

... 0000 (пустое начало ленты)
1011 (двоичное представление одиннадцати)
111110 (обозначает окончание числа *n*)
110 (двоичное представление шести)
10000... (остаток ленты)

То, что машина Тьюринга *U* должна была бы делать на каждом очередном шагу процедуры, выполняемой T_n над *m* – это исследовать структуру последовательности цифр в выражении *n* с тем, чтобы можно было произвести соответствующие изменения цифр числа *m* (т.е. «ленты» машины T_n). В принципе, реализация такой машины не вызывает существенных затруднений (хотя и довольно громоздка на практике). Список ее собственных команд должен был бы просто содержать правила для чтения подходящей команды из «списка», закодированного в числе *n*, на каждом этапе выполнения действий над цифрами, считанными с «ленты», как они фигурируют в числе *m*. Можно предположить, что при этом совершалось бы значительное количество прыжков взад–вперед по ленте между цифрами, составляющими *n* и *m*, и выполнение процедуры было бы чрезвычайно медленным. Тем не менее, список команд подобной машины, несомненно, можно составить, и такая машина называется нами универсальной машиной Тьюринга. Обозначая ее действие на пару чисел (*n*, *m*) через $U(n,m)$, мы получаем:

$$U(n,m) = T_n(m)$$

при любых (*n*, *m*), для которых T_n – корректно определенная машина Тьюринга.[128] Машина *U*, в которую первым вводится число *n*, в точности имитирует *n*-ю машину Тьюринга!

Поскольку *U* – машина Тьюринга, то она сама будет иметь номер. То есть, для некоторого числа *u* имеем

$$U = T_u.$$

Сколько велико *u*? В сущности, мы можем положить, что *u* в точности равно следующему числу:

u = 7244855335 339317577198 395039615711 237952360672 556559631108 144796606505 059404241090 310483613632 359365644443 458382226883 278767626556 144692814117 715017842551 707554085657 689753346356 942478488597 046934725739 988582283827 795294683460 521061169835 945938791885 546326440925 525505820555 989451890716 537414896033 096753020431 553625034984 529832320651 583047664142 130708819329 717234151056 980262734686 429921838172 157333482823 073453713421 475059740345 184372359593 090640024321 077342178851 492760797597 634415123079 586396354492 269159479654 614711345700 145048167337 562172573464 522731054482 980784965126 988788964569 760906634204 477989021914 437932830019 493570963921 703904833270 882596201301 773727202718 625919914428 275437422351 355675134084 222299889374 410534305471 044368695876 405178128019 437530813870 639942772823 156425289237 514565443899 052780793241 144826142357 286193118332 610656122755 531810207511

[128] Если T_n определена некорректно, то *U* будет действовать так, как если бы число, отвечающее *n*, обрывалось сразу по достижении последовательности из четырех или более единиц в двоичной записи *n*. Остаток выражения будет считан уже как число *m*, после чего устройство начнет совершать некие бессмысленные вычисления! От этого свойства можно при желании избавиться, если представлять *n* в расширенной двоичной форме. Я решил не делать этого, чтобы еще больше не усложнять описание несчастной универсальной машины Тьюринга!

085337633806 031082361675 045635852164 214869542347 187426437544 428790062485
827091240422 076538754264 454133451748 566291574299 909502623009 733738137724
162172747723 610206786854 002893566085 696822620141 982486216989 026091309402
985706001743 006700868967 590344734174 127874255812 015493663938 996905817738
591654055356 704092821332 221631410978 710814599786 695997045096 818419062994
436560151454 904880922084 480034822492 077304030431 884298993931 352668823496
621019471619 107014619685 231928474820 344958977095 535611070275 817487333272
966789987984 732840981907 648512726310 017401667873 634776058572 450369644348
979920344899 974556624029 374876688397 514044516657 077500605138 839916688140
725455446652 220507242623 923792115253 181625125363 050931728631 422004064571
305275802307 665183351995 689139748137 504926429605 010013651980 186945639498
(или какому-нибудь другому подходящему, не менее внушительному по величине числу). Это число, без сомнения, выглядит устрашающе большим! Оно, действительно, чрезвычайно велико, но я не вижу способа, как его можно было бы сделать меньше. Процедуры кодирования и определения, использованные мною для машин Тьюринга, вполне разумны и достаточно просты, и всё же с неизбежностью приводят к подобным несуразно большим числам для реальной универсальной машины Тьюринга.[129]

[129] Я благодарен Давиду Дойчу за то, что он нашел десятичную форму двоичного представления *u*, которое я привожу ниже. Я признателен ему также за проверку того факта, что это двоичное значение *u* действительно задает универсальную машину Тьюринга! Двоичная запись *u* выглядит следующим образом: 1000000 00101110 10011010 00100101 01011010 00110100 01010000 01101010 01101000 10101001 01101000 01101000 10100101 01101001 00111010 01010010 01011101 01000111 01010100 10010101 11010101 00110100 01010001 01011010 00001101 00100000 10101101 00010011 10100101 00001010 11101001 00011101 00101010 00010111 01001010 01101000 01000011 10101000 01110101 00001001 00111010 00101010 11010100 10101101 00000110 10101001 01101001 00100011 01000000 00110100 00001110 10100101 01010111 01000010 01110100 10101010 10101011 10100001 01010111 01000010 10001011 10100010 10011010 01000010 10011010 01010010 01101001 00010110 10100010 11101001 00101011 10100101 00011101 01001010 01001110 10101010 00011010 01010101 01110101 00100010 11010100 00101101 01000100 11010101 01010001 01101001 01010010 01011010 10010010 11101010 10010101 11010100 10100110 10101000 01110100 01001001 01011101 01010010 10111010 10100000 11101010 01000001 10101010 10010111 01010010 10110100 01001000 11101000 00001110 10010100 10101010 11101001 01001001 01011101 00000101 01110100 00100011 10100000 10101001 11010000 10100111 01000001 00010111 01000100 00111010 00010010 10011101 00010000 10110100 01010010 11101000 10100101 10100100 00010110 10001010 10010011 01000101 01010111 01001000 00111010 01001010 10101110 10101010 01101001 00010101 10100100 10010110 10000000 10110100 00010001 10100000 10010110 10000000 00110100 10100010 11101001 01010001 10100101 00101011 01000001 00111010 01010100 10110100 10011101 01000000 10101110 10100000 01101010 10001010 10110100 10101011 01010000 10101110 10100100 10101110 10100010 01011010 10010000 10111010 00000111 01010010 00101101 01001010 01101010 10001011 10101001 01001011 10101010 00001011 10101010 00001011 10100000 01110101 01000010 10111010 01010101 10101010 00010111 01010001 01010111 01010100 10010111 01010101 00001110 10100000 00111010 01001000 01101001 00100010 11010101 01010011 10100000 00010110 10010000 11010101 01010010 11101001 00001101 00100010 10101110 10000100 01110100 01000011 10100001 10100000 00101101 00000100 10111010 10100101 01011010 00100010 01011101 00000100 11101010 10011010 00001010 10110100 00100001 11010010 00010001 11010101 01010100 11101000 01001001 11010001 00100001 11010000 10100101 10100001 01000011 10101010 10101011 10100010 01001101 00010010 01101010 01010010 11101000 10001010 11101000 00001110 10001001 00101110 10011010 01001000 01011010 10101001 10100010 10001011 10100001 10101000 01000101 10101001 10101010 01010010 11010101 00110100 10010101 11010011 01001000 00101101 00010101 01000111 01001000 01010110 10000001 00110100 10001001 01110100 10000110 10100000 10010111 01001001 01001101 00100101 01011010 01101001 00101001 01101001 10100101 00000101 10100100 00011101 01001001 10101010 10000101 11010010 10000101 11010010 10101011 10101000 10010110 10010011 10100101 01000101 11010001 00111010 10000101 10100100 11101001 01010101 01110100 10001110 10010101 01001011 10100100 01110101 00000101 01011100 11010100 00010110 10010011 10101000 00010111 01001011 01010000 01010110 10010100 10111010 10000100 10111010 00011010 10001000 01011010 10011010 10001000 10110101 01010010 11101010 00101001 01101000 10101010 11101001 00001010 11010100 01011101 01001001 01010111 01010100 10010111 01010001 11010100 01110101 00100100 10111010 10001110 10100101 00010111 01010001 01110101 00001001 01110101 00011101 00010100 01011101 00101001 01110101 00101010 01011101 00101010 10101011 01010000 10101010 11010000 10011101 00001010 10101011 10101010 00101011 10101010 00101011 10100000 01110101 01000100 10111010 00000111 01010100 10001011 10101000 00011010

Я уже говорил, что все современные общеупотребительные компьютеры, по сути, являются универсальными машинами Тьюринга. Я ни в коем случае не подразумеваю под этим, что их логическая структура должна в точности походить на предложенную мной выше структуру универсальной машины Тьюринга. Однако суть дела состоит в том, что если сперва ввести в произвольную универсальную машину Тьюринга соответствующую программу (начало подаваемой на вход ленты), то потом она сможет копировать поведение любой машины Тьюринга! В предыдущем примере программа просто принимает форму одного числа (числа *n*), но этим разнообразие возможных процедур и вариантов исходной схемы Тьюринга отнюдь не исчерпывается. В действительности я сам, описывая машину, несколько отклонился от того, что исходно было предложено Тьюрингом. Но ни одно из этих отклонений не имеет сейчас для нас существенного значения.

§2.7. Неразрешимость проблемы Гильберта

Мы теперь вплотную подходим к той цели, ради которой Тьюринг с самого начала разрабатывал свою теорию – получить ответ на вопрос, заключенный в общей проблеме алгоритмической разрешимости, поставленной Гильбертом, а именно: существует ли некая механическая процедура для решения всех математических задач, принадлежащих к некоторому широкому, но вполне определенному классу? Тьюринг обнаружил, что он мог бы перефразировать этот вопрос следующим образом: остановится ли в действительности *n*-я машина Тьюринга, если на ее вход

10000101 10100000 01110100 10000001 01110101 00011101 01001000 10101110 10100110 10101010 00101011 01000001 10101010 10010101 01101000 00010011 01010101 00100111 01010011 01010101 00100101 10101001 10100100 10011101 00000110 10101010 10010101 10101000 10011010 00101001 01010111 01000001 10101010 10101001 01101000 10001110 10001010 10101010 11010001 00011101 00001010 11101000 10010000 11101001 10100000 00100111 01000000 10010111 01000100 01010011 10100000 01001011 10100101 01010100 10110100 00101010 10111010 00100101 00101110 10000010 00101110 10101001 01101000 10001001 11010000 01001010 11101000 00010101 01101000 01000111 00111101 00001000 00111010 00010010 01110100 00010100 10111010 00001010 01011010 00010001 01011101 00001000 10011010 00100001 11010111 10100001 00100101 11010000 10010010 11101000 00001010 11101000 01010100 01101000 10010111 01000010 00001110 10000100 11101000 10000010 11101010 10010110 10001000 00101110 10000101 01010111 01000000 10101011 10100010 00010101 11010001 00001010 11101001 00000111 01010010 01001101 00000010 10111010 00100010 01011101 01010000 11101010 01010110 10010101 01000011 01000001 01001101 00000001 11010000 01001001 11010010 11010010 00101001 01101010 10011010 00101001 00101101 01010011 01000101 01000101 10011010 10010010 11101010 10011010 00101010 10101100 11010100 01010101 10011010 01000101 01010111 01000100 01110100 10010101 01010110 10010100 10100011 01001000 00010111 01000001 10101010 01010101 01101001 01010110 10010001 00010111 01000101 01011010 10000010 10110100 01000001 10100100 01010110 10000100 11101010 01010101 01011101 00101101 00100100 01010110 01101001 00100101 01011101 00110100 10010010 10110100 10110100 10010010 01011010 01011010 01001010 00101100 11010010 01010010 10111010 00101011 10100100 10111001 10100100 10101001 01110011 01001010 00101010 11101000 10001110 10000101 00101101 00101000 10111010 01010001 01011010 00100111 01001010 00100101 11010001 00111010 01010010 00101110 01101001 00010001 11010001 00111010 01010010 10101110 01101001 01000001 11001101 01010101 01101000 00001110 10010101 00101010 11101001 00011101 00101010 01010111 00110100 00101001 00110011 01010000 01101000 00001110 10010101 01001010 11100110 10100010 00011010 00000011 10100010 01010101 01110100 01000111 01010101 01010101 01101000 01001110 10010001 00101011 10100101 01000100 11010100 00000101 10100100 11101010 00010101 11010010 00011010 10000000 10110100 10001110 10100100 10111010 00011010 10000101 01011010 10001011 10101000 01010010 11101010 00101110 10100010 10101011 10011010 10001010 11010000 11010100 01001010. Пытливый читатель, вооруженный эффективным домашним компьютером, быть может захочет проверить, используя данные в тексте предписания и применяя эту последовательность к номерам различных простых машин Тьюринга, что она и в самом деле соответствует универсальной машине Тьюринга! Некоторое уменьшение величины *u* может быть достигнуто за счет другого определения машины Тьюринга. Например, мы могли бы отказаться от использования команды STOP и вместо этого применять правило остановки после того, как машина повторно возвращается во внутреннее состояние 0 из какого-либо другого внутреннего состояния. Это не дало бы нам значительного выигрыша (а может, и вовсе никакого). Большую пользу принесло бы использование лент с иными, нежели только 0 и 1, отметками. В литературе встречаются описания очень компактных на вид машин Тьюринга, но эта компактность обманчива, поскольку она обусловлена чрезмерно сложным кодированием описаний машин Тьюринга вообще.

поступит число m? Эта задача получила название проблемы остановки. Не так сложно составить список команд, для которых машина никогда не остановится при любом m (как, например, в случаях $n = 1$ или 2, рассмотренных в предыдущем разделе, а также во всех случаях, когда вообще отсутствует команда STOP). Точно так же существует множество списков команд, для которых машина будет останавливаться всегда, независимо от вводимого числа m (например, T_{11}). Кроме того, некоторые машины при работе с одними числами останавливались бы, а с другими – нет. Совершенно очевидно, что алгоритм, который никогда не прекращает работу, бесполезен. Это, собственно, и не алгоритм вовсе. Поэтому важно уметь ответить на вопрос, приведет ли когда-нибудь работа машины T_n над данным числом m к какому-то ответу или нет! Если нет (т.е. процесс вычисления никогда не прекращается), то я буду выражать это следующей записью:

$$T_n(m) = \Box.$$

(Сюда же включены машины, которые в ходе работы попадают в ситуацию, когда нет команды, определяющей их дальнейшее поведение, как это было в случае рассмотренных выше фиктивных машин T_4 и T_7. К сожалению, наша на первый взгляд работоспособная машина T_3 должна теперь также считаться фиктивной, т.е. $T_3(m) = \Box$, поскольку результатом ее действия всегда будет просто пустая лента, тогда как нам, чтобы приписать номер полученному ответу, нужна хотя бы одна единица на выходе! Машина T_{11}, однако, совершенно полноправна, поскольку она производит единственную **1**. Результатом ее работы будет лента с номером 0, так что $T_{11}(m) = 0$ для любого m.)

В математике весьма важно иметь возможность установить момент, когда машина Тьюринга остановится. Рассмотрим для примера уравнение

$$(x + 1)^{w+3} + (y + 1)^{w+3} = (z + 1)^{w+3}.$$

(Не пугайтесь, даже если вы не любите вникать в детали математических вычислений. Это уравнение используется здесь только в качестве примера, и от вас не требуется его глубокого понимания.) Это конкретное уравнение относится к известной (возможно, самой известной) и пока нерешенной математической проблеме. Проблема формулируется следующим образом: существует ли какой-либо набор x, y, z, w, для которого это равенство выполняется. Знаменитое утверждение, записанное на полях «Арифметики» Диофанта великим французским математиком семнадцатого столетия Пьером де Ферма (1601–1665) и известное как «последняя теорема Ферма», гласит, что это равенство никогда не выполняется[130].[131] Будучи адвокатом по профессии, Ферма тем не менее был искуснейшим математиком своего времени. (Ферма был современником Декарта.) В своей записи он утверждал, что знает «воистину прекрасное доказательство» своей теоремы, но поля книги слишком малы, чтобы его привести. До сегодняшнего дня никому так и не удалось ни воспроизвести это доказательство,[132] ни найти опровергающий это утверждение пример!

Очевидно, что для заданной четверки чисел (x, y, z, w) выяснить, выполняется это равенство или нет, можно простым вычислением. Значит, мы можем представить себе вычислительный алгоритм, который последовательно перебирает все возможные четверки чисел одну за другой и останавливается только тогда, когда равенство удовлетворяется. (Мы уже знаем, что для конечных наборов чисел существуют способы их кодирования на ленте вычислимым способом, а именно, в виде одного числа. Таким образом, перебор всех четверок можно провести, просто следуя естественному порядку соответствующих им одиночных чисел.) Если бы мы могли установить, что этот алгоритм никогда не останавливается, то это стало бы доказательством утверждения Ферма.

Сходным образом в терминах проблемы остановки машины Тьюринга можно перефразировать многие другие нерешенные математические проблемы. Примером такого рода проблем может служить так называемое предположение Гольдбаха: любое четное число, большее двух,

---

[130] Напомним, что натуральными мы называем числа 0, 1, 2, 3, 4, 5, 6... Вместо обычной записи ($x^w + y^w = z^w$, где $x, y, z > 0$, $w > 2$) мы используем «$x + 1$», «$w + 3$» и т.д., чтобы включить в рассмотрение все натуральные числа, начиная с нуля.

[131] Желающие ознакомиться с вопросами, имеющими отношение к этому знаменитому утверждению и изложенными без излишних технических подробностей, могут обратиться к работе Дэвлина [1988].

[132] Последняя теорема Ферма доказана английским математиком Эндрю Уайлсом (Andrew J. Wiles). Доказательство опубликовано в 1995 году. – *Прим. ред.*

может быть представлено в виде суммы двух простых чисел.[133] Процесс, с помощью которого можно установить, относится некоторое натуральное число к простым или нет, является алгоритмическим, поскольку достаточно проверить делимость данного числа на все числа, меньшие его, а это достигается с помощью конечного числа вычислительных операций. Мы можем придумать машину Тьюринга, которая перебирает четные числа 6, 8, 10, 12, 14, ..., пробуя все возможные способы разбиения их на пары нечетных чисел

$$6 = 3 + 3,\ 8 = 3 + 5,\ 10 = 3 + 7 = 5 + 5,\ 12 = 5 + 7,\ 14 = 3 + 11 = 7 + 7, \ldots$$

и убеждаясь, что для каждого четного числа какое-то из разбиений образовано двумя простыми числами. (Очевидно, нам не надо проверять пары четных слагаемых, кроме 2 + 2, поскольку все простые числа за исключением 2 – нечетные.) Наша машина должна остановиться только в том случае, если она находит четное число, для которого ни одно из разбиений не является парой простых чисел. В этом случае мы получили бы контрпример к предположению Гольдбаха, т.е. нашли бы четное число, большее 2, которое не является суммой двух простых чисел. Следовательно, если бы мы могли установить, останавливается машина Тьюринга когда-нибудь или нет, то тем самым мы выяснили бы, справедливо предположение Гольдбаха или нет.

Возникает естественный вопрос: каким образом следует определять, остановится какая-то определенная машина Тьюринга (в которую введены конкретные начальные данные) или нет? Для многих машин Тьюринга ответить на этот вопрос нетрудно, но, как мы видели выше, иногда для ответа может потребоваться решение какой-нибудь до сих пор не решенной математической задачи. Так существует ли некая алгоритмическая процедура для решения общей проблемы – проблемы остановки – полностью механическим путем? Тьюринг показал, что такой процедуры на самом деле нет.[134]

В сущности, его доказательство сводилось к следующему. Предположим, наоборот, что указанный алгоритм существует.[135] Тогда существует и некая машина Тьюринга H, которая «решает», остановится ли в конце концов n-я машина Тьюринга, действуя на число m. Условимся, что результатом действия машины H будет лента с номером 0, если n-я машина не останавливается, и с номером 1 в противоположном случае:

$$H(n; m) = \begin{cases} 0, & \text{если } T_n(m) = \square, \\ 1, & \text{если } T_n(m) \\ & \text{останавливается.} \end{cases}$$

Здесь мы могли бы воспользоваться способом кодирования пары (n, m), использованным ранее для универсальной машины Тьюринга U. Однако это привело бы к проблеме технического характера, поскольку при некоторых n (например, $n = 7$) T_n будет определена некорректно, и маркер 111101 будет непригоден для отделения на ленте n от m. Чтобы избежать этой проблемы, будем полагать, что n представлено не в двоичной, а в расширенной двоичной форме, тогда как для m будет по-прежнему использоваться обычная двоичная запись. В этом случае комбинации 110 будет достаточно для разделения n и m. Использование точки с запятой в обозначении $H(n; m)$ в отличие от запятой в обозначении универсальной машины $U(n, m)$ указывает на это различие в кодировании.

Представим себе теперь бесконечную таблицу, в которую включены окончательные результаты действий всех возможных машин Тьюринга на все возможные (различные) входные

---

133 Напомним, что простые числа 2, 3, 5, 7, 11, 13, 17, ... – это такие натуральные числа, которые делятся только на самих себя и на единицу. Ни нуль, ни единица простыми числами не считаются.

134 **В.Э.:** То, что такой процедуры нет, не вызывает никаких сомнений с точки зрения программиста. Но вот способ, которым Тьюринг пытается это доказать, и те интерпретации, которые всему этому дает Пенроуз, вызывают существенные возражения. Диагональный метод, который Пенроуз чуть ниже назовет «остроумным и мощным приемом», славится своей путаницей в понятиях и на самом деле никогда ничего не доказывает. Диагональный процесс можно пускать всегда, когда перед нами бесконечное множество, элементы которого тоже бесконечны, – и ничего другого, кроме этого факта, этот «остроумный и мощный прием» не показывает.

135 Это хорошо известный и очень мощный метод математического доказательства, называемый «доказательством от противного» или *reductio ad absurdum* (сведение к абсурду), в котором сначала полагается истинным утверждение, исключающее исходное, затем из этой предпосылки выводится противоречие, которое и служит доказательством справедливости исходного утверждения.

данные. В этой таблице *N*-й ряд представляет собой результаты вычислений *n*-й машины Тьюринга, полученные при ее работе последовательно с *m* = 0, 1, 2, 3, 4, ...:

$n \downarrow$ \ $m \rightarrow$	0	1	2	3	4	5	6	7	8	...
0	□	□	□	□	□	□	□	□	□	...
1	0	0	0	0	0	0	0	0	0	...
2	1	1	1	1	1	1	1	1	1	...
3	0	2	0	2	0	2	0	2	0	...
4	1	1	1	1	1	1	1	1	1	...
5	0	□	0	□	0	□	0	□	0	...
6	0	□	1	□	2	□	3	□	4	...
7	0	1	2	3	4	5	6	7	8	...
8	□	1	□	□	1	□	□	□	1	...
.	.				.				.	...
.	.				.				.	
.	.				.				.	
197	2	3	5	7	11	13	17	19	23	...
.	.				.				.	
.	.				.				.	
.	.				.				.	

Я немного «сжульничал» и не стал располагать машины Тьюринга по порядку их <u>действительных</u> номеров. Если бы я так сделал, то получился бы список, начало которого выглядело бы слишком скучным, поскольку все машины при значениях *n* меньших 11 не дают ничего, кроме □, а для *n* = 11 мы имеем просто нули. Дабы сделать начало этой таблицы более интересным, я предположил, что мы использовали некую гораздо более эффективную систему кодирования. Фактически, я просто присвоил ячейкам более или менее произвольные значения, только чтобы дать вам общее представление о том, как может выглядеть эта таблица.

На самом деле нам не требуется, чтобы эта таблица была построена путем <u>вычислений</u>, скажем, с помощью некоторого алгоритма. (На самом деле, как мы увидим далее, такого алгоритма и не существует.) Достаточно просто <u>представить</u> себе, что каким-то образом <u>истинный</u> список попал в наше распоряжение, возможно, с помощью Бога[136]! Если бы мы попытались получить эту таблицу с помощью вычислений, то именно символы □ вызвали бы затруднения, поскольку мы не могли бы с уверенностью сказать, когда в той или иной ячейке должен быть помещен символ □ – ведь соответствующие вычисления никогда не заканчиваются!

Тем не менее искомую таблицу <u>можно</u> построить с помощью вычислительной процедуры, если использовать нашу гипотетическую машину *H*, поскольку она могла бы определить, где на самом деле появляются значения □. Однако вместо этого мы используем машину *H* для того, чтобы <u>избавиться</u> от появления значений □ в таблице, заменив их во всех случаях нулями. Это достигается за счет вычисления значения $H(n;\, m)$, предваряющего действие T_n на *m*, после чего мы позволим T_n производить соответствующие действия, только если $H(n;\, m) = 1$ (т.е. только тогда, когда вычисление $T_n(m)$ приводит к определенному результату), и будем просто записывать в соответствующую ячейку 0 при $H(n;\, m) = 0$ (т.е. если $T_n(m) = \square$). Мы можем записать эту новую процедуру, представляющую собой последовательное действие $H(n;\, m)$ и $T_n(m)$, как

$$T_n(m) \times H(n;\, m).$$

(Здесь я использую общепринятую в математике договоренность о последовательности выполнения действий, согласно которой операция, записанная <u>справа</u>, должна выполняться <u>первой</u>. Обратите внимание, что в этом случае можно символически записать $\square \times 0 = 0$.)

Теперь таблица принимает следующий вид:

[136] **В.Э.:** Обозначим эту «данную Богом» матрицу как *M.

$n \downarrow$ \ $m \rightarrow$	0	1	2	3	4	5	6	7	8	…
0	0	0	0	0	0	0	0	0	0	…
1	0	0	0	0	0	0	0	0	0	…
2	1	1	1	1	1	1	1	1	1	…
3	0	2	0	2	0	2	0	2	0	…
4	1	1	1	1	1	1	1	1	1	…
5	0	0	0	0	0	0	0	0	0	…
6	0	0	1	0	2	0	3	0	4	…
7	0	1	2	3	4	5	6	7	8	…
8	0	1	0	0	1	0	0	0	1	…
.	.				.				.	
.	.				.				.	
.	.				.				.	

Заметьте, что, исходя из предположения существования машины H, мы получаем ряды таблицы, состоящие из вычислимых последовательностей. (Под «вычислимой последовательностью» я понимаю бесконечную последовательность, элементы могут быть найдены один за другим посредством некоего алгоритма; это означает, что существует некоторая машина Тьюринга, которая, будучи применена поочередно к натуральным числам $m = 0, 1, 2, 3, 4, 5, ...$, производит члены рассматриваемой последовательности.) Обратите внимание на следующие два факта относительно этой таблицы. Во-первых, любая вычислимая последовательность натуральных чисел должна появиться где-то (может быть, далеко не сразу) среди рядов таблицы.[137] Это свойство выполнялось уже и для исходной таблицы, содержавшей значения □. Мы просто добавили несколько рядов, чтобы заменить «фиктивные» машины Тьюринга (т.е. такие, которые приводят к □ хотя бы в одном случае). Во-вторых, считая, что машина Тьюринга H существует,

[137] **В.Э.:** Ну вот, здесь и раскрывается вся ущербность всех этих построений! Допустим, как и в {PENRS1} §2.5, что «в нашей вселенной» всего пять чисел (0, 1, 2, 3, 4), т.е. m может принимать только эти значения. Тогда одних только перестановок P_m этих чисел (а они все вычислимы и, значит, должны находиться в строках пенроузовской таблицы!) будет 5! (факториал от 5, т.е. 120 штук!). А плюс еще случаи, когда элементы последовательности повторяются, а плюс еще когда они все одинаковы как в пенроузовской строке 2 (одних этих будет столько же, сколько и m!), а плюс еще когда разными машинами Тьюринга выдаются одинаковые последовательности, как в пенроузовских строках 0 и 1... Эта таблица (если она претендует на то, что содержит все вычислимые последовательности) никогда не будет квадратной – вниз она намного намного длиннее, чем вправо. И при росте m, и при $m \to \infty$ всё это будет только усугубляться. Следовательно, диагональный процесс никогда не охватит все строки таблицы. Он упрется в правый край таблицы, не достигнув ее нижнего края. И построенный диагональным процессом новый элемент будет в таблице (только в неохваченной диагональным процессом ее части), и никакого противоречия получено не будет, и всё «доказательство» Тьюринга–Пенроуза полетит к чертовой матери... (Всё как обычно при диагональном процессе – надоело уже и разбирать все эти вариации). Но зато пенроузовская процедура Q не сможет обработать эту таблицу и умножить $T_n(m) \times H(n; m)$. Она не может сначала обработать первую строку, уйти в бесконечность, перепрыгнуть через нее, потом взяться за вторую строку, уйти во вторую бесконечность, снова перепрыгнуть через нее и т.д. Она может обрабатывать таблицу только «с уголка»: сначала взять элемент n=0, m=0; потом n=0, m=1; потом n=1, m=1; потом n=1, m=0; потом n=0, m=2; потом n=1, m=2; потом n=2, m=2; потом n=2, m=1; потом n=2, m=0 и т.д. Но обработанная таким образом матрица будет квадратной. Она не охватит всю исходную таблицу, и если в ней (в этой квадратной обработанной матрице) пускать диагональный процесс, то он действительно охватит все ее строки и построит элемент, в ней не содержащийся. Но никакого противоречия всё равно не будет, потому что эта квадратная обработанная процедурой Q матрица (в отличие от исходной вытянутой) действительно не содержит все последовательности. Из всего этого, разумеется, не следует, что процедура $H(n; m)$ существует. Она не существует, но просто все эти путанные «рассуждения» не доказывают этого. В моем старом учебнике логики (Г.И. Челпанов, 1947), по которому во времена моего младенчества мой отец преподавал логику в гимназии, эта ситуация называется «утверждение формально неправильное, но материально правильное». Однако больше всего меня удивляет то, что всю эту белиберду с чрезвычайно путанными понятиями Пенроуз (и другие математики!) могут воспринимать всерьез и делать из этого какие-то выводы, относящиеся к нашему реальному миру. Это же какое-то наваждение, гипноз, помешательство!

мы получили таблицу вычислительным путем (т.е. с помощью некоторого определенного алгоритма), а именно, посредством процедуры $T_n(m) \times H(n;\ m)$. Иными словами, существует некая машина Тьюринга Q, применение которой к паре чисел $(n,\ m)$ дает значение соответствующей ячейки таблицы. Для этой машины числа n и m на ленте можно кодировать таким же образом, как и для H, т.е. мы имеем

$$Q(n;\ m) = T_n(m) \times H(n;\ m).$$

Воспользуемся теперь разновидностью остроумного и мощного приема, так называемого диагонального процесса Георга Кантора. (Мы познакомимся с оригинальным вариантом этого метода в следующей главе.) Рассмотрим значения в ячейках, расположенных на главной диагонали таблицы – диагональные элементы (матрицы), – выделенные **жирным** шрифтом:

$$\begin{array}{ccccccccc}
\mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \mathbf{0} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & \mathbf{1} & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 2 & 0 & \mathbf{2} & 0 & 2 & 0 & 2 & 0 \\
1 & 1 & 1 & 1 & \mathbf{1} & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & \mathbf{0} & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 2 & 0 & \mathbf{3} & 0 & 4 \\
0 & 1 & 2 & 3 & 4 & 5 & 6 & \mathbf{7} & 8 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & \mathbf{1} \\
\cdot & & & & \cdot & & & & \cdot \\
\cdot & & & & \cdot & & & & \cdot \\
\cdot & & & & \cdot & & & & \cdot
\end{array}$$

Эти элементы образуют некоторую последовательность 0, 0, 1, 2, 1, 0, 3, 7, 1, ..., к каждому члену которой мы теперь прибавим единицу:

$$1, 1, 2, 3, 2, 1, 4, 8, 2, \ldots .$$

Это, безусловно, механическая процедура, и, поскольку наша таблица была получена путем вычислений, мы получим новую вычислимую последовательность $1 + Q(n;\ n)$, т.е.

$$1 + T_n(n) \times H(n;\ n)$$

(с учетом того, что для диагональных элементов $n = m$). Но наша таблица содержит в себе все вычислимые последовательности, поэтому она должна содержать также и новую последовательность. Однако это невозможно! Ведь наша новая последовательность отличается от первого ряда первым элементом, от второго – вторым, от третьего – третьим, и т.д. Налицо явное противоречие, которое и устанавливает справедливость доказываемого нами утверждения о том, что машина Тьюринга H на самом деле не существует! Иными словами, не существует универсального алгоритма для решения вопроса об остановке произвольной машины Тьюринга.

Можно построить доказательство и по-другому. Для этого заметим, что из предположения о существовании H следует и существование машины Тьюринга с номером k, реализующей алгоритм (диагональный процесс!) $1 + Q(n;\ n)$, т.е. можно записать

$$1 + T_n(n) \times H(n;\ n) = T_k(n).$$

Но если мы подставим в это выражение $n = k$, то получится

$$1 + T_k(k) \times H(k;\ k) = T_k(k).$$

Мы приходим к противоречию, потому что если $T_k(k)$ останавливается, то мы имеем невыполнимое равенство

$$1 + T_k(k) = T_k(k)$$

(поскольку $H(k;\ k) = 1$), тогда как в случае безостановочного действия $T_k(k)$ (т.е. когда $H(k;\ k) = 0$) мы получаем не менее абсурдное соотношение

$$1 + 0 = \square.$$[138]

Вопрос о том, останавливается ли конкретная машина Тьюринга или нет, представляет собой совершенно четко определенную математическую задачу (а ранее мы уже видели, что, наоборот, различные важные математические задачи могут быть сведены к вопросу об остановке машины Тьюринга). Таким образом, доказав, что не существует алгоритма для решения вопроса

---

[138] **В.Э.:** Эти противоречия доказывают только лишь то, что диагональ $Q(n;\ n)$ не может пересекаться с прямой T_k, как это и должно быть при вытянутой вниз матрице и при том, что T_k намеренно строит нечто отличное от того, что содержится в области, охваченной диагональю. Загипнотизированный Кантором и Тьюрингом Пенроуз не понимает, что полученным противоречием опровергается не существование $H(n;\ m)$, а существование $T_k(k)$.

об остановке машины, Тьюринг показал (также как и Черч, который использовал свой собственный и весьма отличающийся подход), что не может быть и общего алгоритма для решения математических задач. Проблема разрешимости Гильберта не имеет решения!

Это не означает, что в каждом отдельном случае мы не в состоянии выяснить справедливость (или, наоборот, несостоятельность) некоторого конкретного математического утверждения или определить, остановится ли данная машина Тьюринга. С помощью интуиции, искусных технических приемов или же опираясь просто на здравый смысл, мы, вероятно, могли бы получить ответ на такие вопросы в частных случаях. (Так, например, если перечень инструкций некоторой машины Тьюринга не включает ни одной команды STOP или же, наоборот, состоит только из таких команд, то одного здравого смысла[139] достаточно для решения вопроса о ее остановке!) Но не существует ни одного алгоритма, который позволял бы решать любую математическую задачу или давал ответ на вопрос об остановке любой машины Тьюринга при любых вводимых в нее числах.

Может показаться, что мы пришли к выводу о существовании по крайней мере нескольких неразрешимых математических вопросов. Однако это совсем не так! Мы не показали, что существует какая-то необычайно громоздкая машина Тьюринга, для которой (в некотором абсолютном смысле) невозможно решить вопрос об остановке при ее работе с каким-то особенно громоздким числом – в действительности, всё как раз наоборот, как мы сможем скоро убедиться. Мы вообще ничего не говорили о неразрешимости какой-то отдельной задачи, а только лишь об алгоритмической неразрешимости классов задач. В каждом конкретном случае ответ будет либо «да», либо «нет», поэтому алгоритм для решения частной задачи, конечно, существует, а именно алгоритм, который при применении к этой задаче просто дает ответ «да» или, может быть, «нет»! Трудность в данном случае состоит в том, что мы не знаем, какой именно из имеющихся алгоритмов применять в том или ином случае. Это вопрос об установлении математической истинности отдельного утверждения, но не об общем решении проблемы для целого класса утверждений. Очень важно сознавать, что сами по себе алгоритмы не доказывают математическую истину.[140] Решение о правомерности использования каждого алгоритма должно всегда приходить извне.

§2.8. Как превзойти алгоритм

К вопросу о том, как установить истинность математических утверждений, мы вернемся позднее, в связи с теоремой Гёделя (см. главу 4). Пока же я бы хотел обратить ваше внимание на то, что доказательство Тьюринга носит гораздо более конструктивный характер и не столь негативно, как могло показаться из предыдущего изложения. Мы ведь не показали, что есть некая определенная машина Тьюринга, для которой абсолютно невозможно решить, останавливается она или нет. Более того, если внимательно проследить за доказательством, то выяснится, что для кажущихся «чрезвычайно сложными» машин сама процедура Тьюринга, использованная для их построения, неявным образом дает ответ! Посмотрим, как это происходит. Допустим, у нас есть алгоритм, который иногда позволяет определить, что машина Тьюринга не остановится. Вышеописанная процедура Тьюринга позволяет явно проследить за вычислениями машины Тьюринга в случае, когда этот конкретный алгоритм не дает ответа на вопрос об остановке вычислительного процесса. Однако тем самым эта процедура дает нам в этом случае возможность узнать ответ! Конкретная машина Тьюринга, за работой которой мы следим, и вправду никогда не остановится.

Чтобы подробно разобраться в этом вопросе, предположим, что у нас есть некий алгоритм, который иногда позволяет решить проблему остановки. Как и ранее, мы обозначим этот алгоритм (машину Тьюринга) через H, но теперь мы допускаем, что этот алгоритм не всегда может точно определить, что машина Тьюринга не остановится:

[139] **В.Э.:** Однако этот «здравый смысл» опять же есть не что иное, как алгоритм («проверить, есть ли команды STOP в тексте программы (машины Тьюринга)» и т.п.).

[140] **В.Э.:** Это уже общие и неправильные интерпретации всего предыдущего. Если «математическую истину» вообще можно установить, то только по тому или иному алгоритму.

$$H(n;m) = \begin{cases} 0 \text{ или } \square, & \text{если } T_n(m) = \square, \\ 1, & \text{если } T_n(m) \\ & \text{останавливается.} \end{cases}$$

так что $H(n; m) = \square$ возможно в случае, когда $T_n(m) = \square$. Существует немало алгоритмов типа $H(n; m)$. (Например, $H(n; m)$ мог бы просто давать на выходе 1, как только машина $T_n(m)$ останавливается, хотя такой алгоритм едва ли представляет большой практический интерес!)

Мы можем повторить процедуру Тьюринга, следуя уже пройденным путем, с той только разницей, что теперь некоторые из «□» останутся не замененными на нули. Как и ранее, применив диагональный процесс, получим

$$1 + T_n(n) \times H(n; n)$$

в качестве n-го элемента диагонали. (Мы будем иметь □ каждый раз, когда $H(n; n) = \square$. Отметим, что $\square \times \square = \square$, $1 + \square = \square$.) Это безупречно алгоритмизованное вычисление, поэтому оно может быть произведено некоторой машиной Тьюринга, скажем k-ой, и тогда мы получим

$$1 + T_n(n) \times H(n; n) = T_k(n).$$

Для k-го диагонального элемента (т.е. $n = k$) мы имеем

$$1 + T_k(k) \times H(k; k) = T_k(k).$$

Если вычисления $T_k(k)$ останавливаются, то мы приходим к противоречию (в этом случае $H(k; k)$ должно равняться единице, но тогда возникнет невыполнимое равенство: $1 + T_k(k) = T_k(k)$). Значит, $T_k(k)$ не может остановиться, т.е.

$$T_k(k) = \square.$$

Но алгоритм не может этого «знать», потому что, если бы он давал $H(k; k) = 0$, мы снова пришли бы к противоречию (мы получили бы тогда неверное соотношение $1 + 0 = \square$).

Таким образом, если мы можем отыскать k, то мы знаем, как построить вычислительную процедуру, для которой алгоритм не дает решения проблемы остановки, но нам ответ известен!

* * *

2010.12.11 22:19 суббота

В.Э.: Так рассуждает Пенроуз. А теперь посмотрим, как выглядят точные рассуждения. Как обычно, будем доверять математической индукции, а не диагональному процессу (как известно, они находятся в непримиримом противоречии – см. {PENRS1} §2.5).

2011.01.05 15:21 среда

Следуя математической индукции, сначала (1) посмотрим, как это всё выглядит при конечном $m = z–1$ ($z–1$ – максимальное значение m, которое может подаваться на вход алгоритмам – машинам Тьюринга T_n – или выдаваться ими), а потом (2) посмотрим, что произойдет, когда $z \to \infty$.

Тогда «данная Богом» матрица *M будет иметь «вправо» ширину z колонок, а «вниз» некоторую длину $\check{z}$ [141] строк. Так как эта матрица *M включает в себе ВСЕ последовательности чисел, какие только могут быть созданы алгоритмическим путем (машинами Тьюринга) из чисел в пределах от 0 до $z – 1$, то очевидно, что $\check{z} >> z$ ($\check{z}$ намного больше z). Одних только перестановок чисел m среди $\check{z}$ строк будет $z!$ (факториал от z). Графически ситуация изображена на Рис. VE3.

Далее, очевидно, что диагональ, по которой будет строиться «отличающаяся от всех существующих» (и якобы вызывающая противоречие) строка k матрицы, – эта диагональ не охватывает все $\check{z}$ строки матрицы, а только z строк. Так как k намеренно строится такой, чтобы она не совпадала ни с одной из строк, охваченных диагональю, то, естественно, строка k находится (если уж матрица вообще содержит все строки!) в той части матрицы, которую «диагональный процесс» не охватил.

Предположение, что существует $T_k(k)$, то есть, что существует результат машины T_k при входном значении k, естественно, должно приводить к противоречиям, потому что $k > z$ (то есть, это значение m недопустимо для принятых нами ограничений) и потому, что диагональ и строка k не пересекаются.

[141] Читается: «ж».

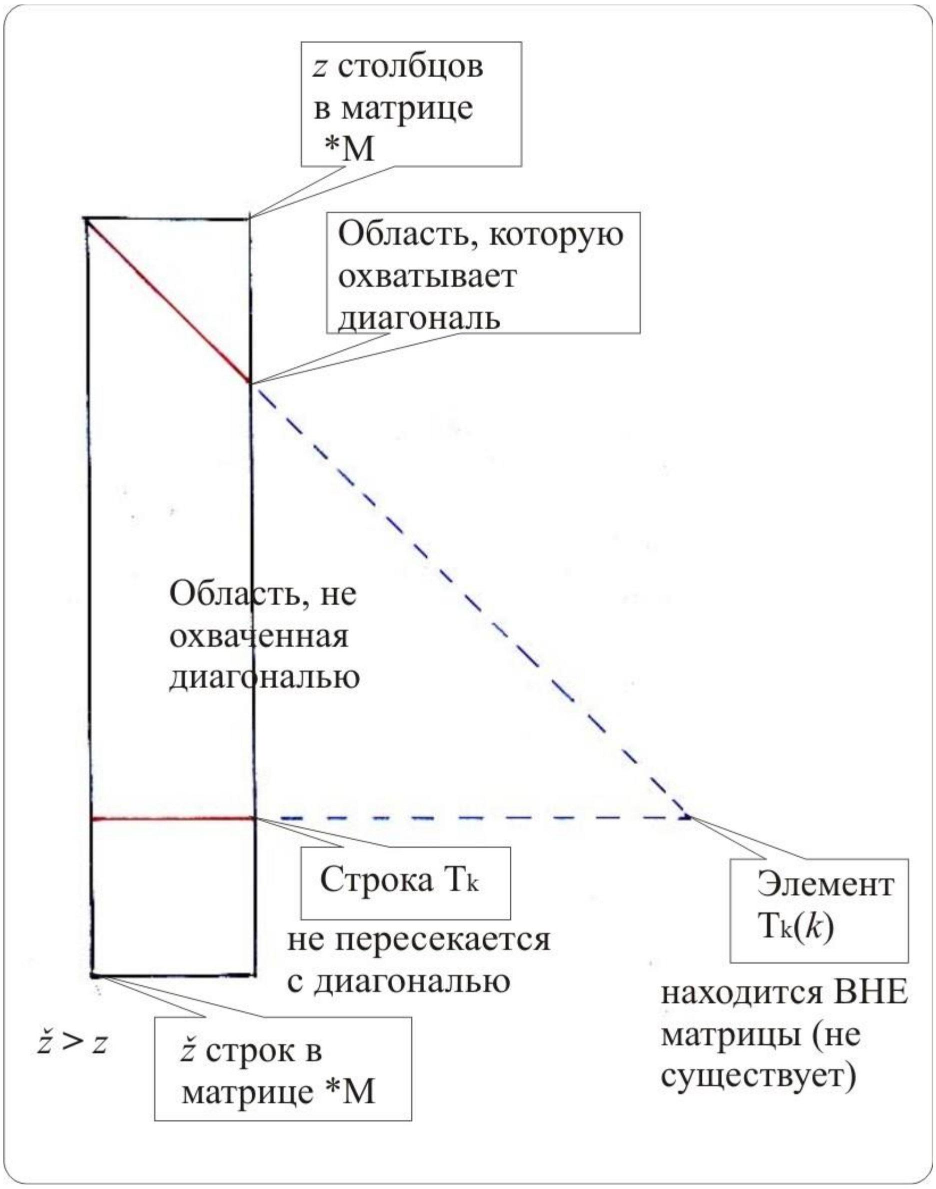


Рис. VE3. Диагональный процесс в матрице *М, данной Богом Пенроузу

Все полученные противоречия доказывают недопустимость предположения, что существует $T_k(k)$, а не предположения, что существует машина $H(n;\ m)$. Всё это очевидно даже из «визуализации» рисунка VE3.

Переносить недопустимость $T_k(k)$ на недопустимость $H(n;\ m)$ – это логическая (и математическая) ошибка, причем чрезвычайно грубая!

Но Пенроуз ее совершает!

А далее пусть $z \to \infty$. Что изменится?

Ничего. Матрица растягивается еще больше, диагональ растет бесконечно вправо и вниз (но всегда гораздо медленнее, чем растягивается матрица), строка k летит вниз, тоже растягиваясь в длину до бесконечности, но, тем не менее, всегда оставаясь в той части матрицы, которую не охватывает диагональ. Элемент $T_k(k)$ никогда не существует, и предположение о его существовании всегда будет приводить к указанным Пенроузом противоречиям, а о существовании машины H всё это не говорит абсолютно ничего.

Можно оценить соотношение роста матрицы вправо и вниз. Количество столбцов растет как $z \to \infty$. Зависимость количества строк от z оценить труднее, так как состав множества машин T_n не очень четко определен, но ясно что в нем есть подмножество простых перестановок z чисел. Оценим хотя бы соотношение количества столбцов с этим подмножеством строк. К какому пределу будет стремиться это соотношение при $z \to \infty$? Это соотношение $z/z! = 1/(z–1)!$

В этой дроби переменная, стремящаяся к бесконечности, даже не остается в числителе, а лишь в одном знаменателе – так что и правило Лопиталя применять незачем. Чему равен предел дроби $1/(z–1)!$, когда $z \to \infty$? А эта дробь характеризует ту часть матрицы, которую будет охватывать диагональный процесс!

Чтобы рассуждения Пенроуза (а перед ним Тьюринга) считать состоятельными, нужно предполагать, что при бесконечном z ситуация изменится: что диагональ вдруг охватит всю матрицу, что $T_k(k)$ вдруг окажется в пределах матрицы в результате пересечения диагонали и строки T_k ... Интересно, как Пенроуз и Тьюринг согласовали бы свое видение ситуации с математической индукцией, с правилом Лопиталя – вообще с духом той, старой и достоверной математики, которая создала дифференциальное и интегральное исчисление?

Но по вопросу такого согласования мне никогда не приходилось слышать ничего другого, кроме тупого утверждения доктора Подниекса {CANTO2.2299 = МОИ № 39}, что правило Лопиталя им с Кантором не нужно.

Итак, всё здесь предельно ясно, и непонятно только одно: как могут математики такого ранга как Пенроуз всего этого не видеть, не понимать и принимать всерьез эту Канторо-Тьюринговскую галиматью!? Какое ослепление должно на них найти, как должен быть затуманен и заворожен их разум, чтобы рассуждать так, как они рассуждают, когда истина совершенно очевидна и кристально ясна!

Так что Пенроузу «превзойти алгоритм» все-таки не удалось.

(Конец вставки; далее продолжается текст Пенроуза)

* * *

А как нам найти k? Это непростая задача. Необходимо тщательно изучить конструкцию $H(n; m)$ и $T_n(m)$ и понять, как в точности действует $1 + T_n(n) \times H(n; n)$ в качестве машины Тьюринга. Затем надо определить номер этой машины, который и есть k. Конечно, это выполнить трудно, но вполне возможно.[142] Из-за этих трудностей вычисление $T_k(k)$ нас бы вовсе не интересовало, не будь она специально предназначена для доказательства неэффективности алгоритма H! Важно то, что мы получили строго определенную процедуру, которая для любого наперед заданного алгоритма H позволяет найти такое k, что для $T_k(k)$ этот алгоритм не может решить проблему остановки, т.е. мы тем самым превзошли его. Возможно, мысль о том, что мы «умнее» каких-то алгоритмов, принесет нам некоторое удовлетворение!

На самом деле, упомянутая процедура настолько хорошо определена, что мы могли бы даже найти алгоритм для нахождения k по заданному H. Поэтому, прежде чем мы «погрязнем» в самодовольстве, мы должны осознать, что этот алгоритм может улучшить H,[143] поскольку он, по сути, «знает», что $T_k(k) = \square$, – или все-таки нет? В предыдущем изложении было удобно использовать антропоморфный термин «знать» по отношению к алгоритму. Однако не мы ли в конечном счете «знаем», тогда как алгоритм просто следует определенным нами правилам? А может быть мы сами просто следуем правилам, запрограммированным в конструкции нашего мозга и в окружающей нас среде? Эта проблема затрагивает не только алгоритмы, но и то, как мы выносим суждения об истинности и ложности. К этим важнейшим проблемам мы вернемся позднее. Вопрос о математической истине (и ее неалгоритмической природе) будет рассмотрен в главе 4. На данный момент мы, по крайней мере, получили некоторое представление о значении слов «алгоритм» и «вычислимость» и достигли понимания некоторых из относящихся к ним вопросов.

§2.9. Лямбда-исчисление Черча

Понятие вычислимости – очень важная и красивая математическая идея. Примечателен также и ее малый возраст в сравнении с другими столь же фундаментальными математическим проблемами: она была впервые выдвинута только в 1930-х годах. Эта проблема имеет отношение

[142] Фактически, самую трудную часть мы уже выполнили, когда построили универсальную машину Тьюринга U, поскольку она позволяет нам записывать $T_n(n)$ как машину Тьюринга, действующую на n.

[143] Мы могли бы, конечно, «обыграть» и этот модифицированный алгоритм, просто за счет повторного применения предыдущей процедуры. Тогда мы сможем использовать эти вновь полученные знания для дальнейшего улучшения алгоритма, который мы, в свою очередь, снова превзойдем; и так далее. Тип рассуждений, в который выливается этот повторяющийся процесс, будет рассмотрен нами в связи с теоремой Гёделя в главе 4 (с. 99).

ко всем областям математики (хотя, справедливости ради, отметим, что большинство математиков пока не часто обращаются к вопросам вычислимости). Сила этой идеи связана отчасти с существованием четко определенных и всё же неразрешимых математических операций (как, например, проблема остановки машины Тьюринга и некоторые другие, которые мы рассмотрим в главе 4). Если бы не было таких невычислимых объектов, то теория алгоритмической разрешимости не представляла бы особого интереса для математики. В конце концов, математики любят головоломки. Задача о разрешимости определенной математической операции может их заинтриговать, особенно потому, что общее решение этой головоломки само по себе алгоритмически не разрешимо.

Следует сделать еще одно замечание. Вычислимость – это по-настоящему «абсолютная» математическая идея. Это абстрактное понятие, которое никак не зависит от какой-либо конкретной реализации в терминах «машин Тьюринга» в том виде, как я их описал выше. Как я уже указывал, нет необходимости придавать какое-либо специальное значение «лентам», «внутренним состояниям» и т.п., характерным для гениального, но тем не менее частного подхода Тьюринга. Существуют также и другие способы выражения идеи вычислимости, причем исторически первым было «лямбда-исчисление», предложенное американским логиком Алонзо Черчем совместно со Стивеном Клини. Процедура, предложенная Черчем, значительно отличалась от метода Тьюринга и была гораздо более абстрактна. Фактически, форма, в которой Черч изложил свою теорию, делала связь между ними и чем бы то ни было «механическим» совсем не очевидной. Главная идея, лежащая в основе процедуры Черча, абстрактна по своей сути – это математическая операция, которую сам Черч назвал «абстрагированием».

Мне кажется, что стоит привести краткое описание схемы Черча не только потому, что она подчеркивает математическую природу идеи вычислимости, не зависящую от конкретного понятия вычислительной машины, но и потому, что она иллюстрирует мощь абстрактных идей в математике. Читатель, не достаточно свободный в математике и не увлеченный излагаемыми математическими идеями как таковыми, скорее всего предпочтет сейчас перейти к следующей главе – и не утратит при этом нить рассуждений. Тем не менее я полагаю, что таким читателям будет небесполезно следовать за мной еще какое-то время и оценить чудесную по своей стройности и продуманности схему Черча (см. Черч [1941]).

В рамках этой схемы рассматривается «универсальное множество» различных объектов, обозначаемых, скажем, символами

$$a, b, c, d, \ldots, z, a', b', \ldots, z', a'', b'', \ldots, z'', a''', \ldots, a'''', \ldots,$$

каждый из которых представляет собой математическую операцию или функцию. (Штрихованные буквы позволяют создавать неограниченные наборы символов для обозначения таких функций.) «Аргументы» этих функций, т.е. объекты, на которые эти функции действуют, в свою очередь являются объектами той же природы, т.е. функциями. Более того, результат действия одной функции на другую (ее «значение») также представляет собой функцию. (Поистине, в системе Черча наблюдается замечательная экономия понятий.) Поэтому, когда мы пишем[144]

$$a = bc,$$

мы подразумеваем, что функция b, действуя на функцию c, дает в результате другую функцию a. В рамках этой схемы нетрудно сформулировать понятие функции двух или более переменных. Если мы хотим представить f как функцию двух переменных, скажем p и q, то мы можем просто написать

$$(fp)q$$

(что есть результат действия функции fp на функцию q). Для функции трех переменных можно использовать выражение

$$((fp)q)r$$

и так далее.

Теперь мы можем перейти к описанию важнейшей операции абстрагирования. Для нее мы будем использовать греческую букву λ (лямбда). Непосредственно за ней будет следовать символ одной из функций Черча, скажем x, который мы будем рассматривать как «фиктивную переменную». Каждое появление x в квадратных скобках, следующих сразу за этим выражением,

[144] В более привычной форме эта запись имела бы вид $a = b(c)$, но эти дополнительные скобки в действительности не нужны, поэтому лучше просто привыкнуть к их отсутствию. Их последовательное использование привело бы к довольно громоздким формулам вида $(f(p))(q)$ и $((f(p))(q))(r)$ вместо $(fp)q$ и $((fp)q)r$, соответственно.

обозначает теперь просто место, куда подставляется всё, что идет за всем этим выражением. Таким образом, когда мы пишем

$$\lambda x.[fx]$$

мы подразумеваем функцию, которая при действии на, например, a имеет значение fa, т.е.

$$(\lambda x.[fx])a = fa$$

Другими словами, $\lambda x.[fx]$ – это просто функция f, т.е.

$$\lambda x.[fx] = f$$

Сказанное выше требует определенного осмысления. Это одна из тех математических тонкостей, которые на первый взгляд кажутся настолько педантичными и тривиальными, что их смысл часто совершенно ускользает от понимания. Рассмотрим пример из знакомой всем школьной математики. Примем за f тригонометрическую функцию – синус угла. Тогда абстрактная функция «sin» будет определяться выражением

$$\lambda x.[\sin x] = \sin.$$

(Не придавайте большого значения тому, что в качестве «функции» x может фигурировать величина угла. Мы скоро увидим, каким образом числа можно иногда рассматривать как функции, а величина угла – это просто число.) До сих пор всё на самом деле тривиально. Однако представим себе, что обозначение «sin» не было изобретено, но нам известно о существовании представления $\sin x$ в форме степенного ряда:

$$x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \dots .$$

Тогда мы могли бы ввести определение

$$\sin = \lambda x.\left[x - \frac{1}{6}x^3 + \frac{1}{120}x^5 - \dots\right].$$

Можно было поступить еще проще и определить, например, операцию «одна шестая куба», для которой не существует стандартного «функционального» обозначения:

$$Q = \lambda x.\left[\frac{1}{6}x^3\right].$$

Тогда, например,

$$Q(a+1) = \frac{1}{6}(a+1)^3 = \frac{1}{6}a^3 + \frac{1}{2}a^2 + \frac{1}{2}a + \frac{1}{6}.$$

К обсуждаемым проблемам большее отношение имеют выражения, составленные просто из элементарных функциональных операций Черча, таких как

$$\lambda f.[f(f)x].$$

Это функция, которая, действуя на другую функцию, скажем g, дает дважды итерированную g, действующую на x

$$(\lambda f.[f(f)x])g = g(gx).$$

Мы могли бы сначала «абстрагироваться» от x и рассмотреть выражение

$$\lambda f.[\lambda f.[f(f)x]],$$

которое можно сократить до

$$\lambda f\mathrm{x}.[f(f)x].$$

Это и есть операция, применение которой к g дает функцию «вторая итерация g». По сути, это та самая функция, которую Черч обозначил номером $\mathbb{2}$:

$$\mathbb{2} = \lambda f\mathrm{x}.[f(f)x],$$

так что $(\mathbb{2}g)y = g(gy)$. Аналогичным образом он определил:

$$\mathbb{3} = \lambda f\mathrm{x}.[f(f(f))x],$$

$$\mathbb{4} = \lambda f\mathrm{x}.[f(f(f(f)))x], \text{ и т.д.,}$$

а также

$$\mathbb{1} = \lambda f\mathrm{x}.[fx] \text{ и } \mathbb{0} = \lambda fx.[x].$$

Видно, что $\mathbb{2}$ Черча больше похоже на «дважды», $\mathbb{3}$ – на «трижды» и т.д. Значит, действие $\mathbb{3}$ на функцию f, т.е. $\mathbb{3}f$ равносильно операции «применить f три раза», поэтому $\mathbb{3}f$ при действии на y превращается в $(\mathbb{3}f)y = f(f(f(y)))$.

Посмотрим, как в схеме Черча можно представить очень простую математическую операцию – прибавление $\mathbb{1}$ к некоторому числу. Определим операцию

$$S = \lambda abc.[b((ab)c)].$$

Чтобы убедиться, что S действительно прибавляет $\mathbb{1}$ к числу в обозначениях Черча, проверим ее действие на $\mathbb{3}$:

$$S\mathbb{3} = \lambda abc.[b((ab)c)]\mathbb{3} = \lambda bc.[b((\mathbb{3})c] = \lambda bc.[b(b(b(bc)))] = \mathbb{4},$$

поскольку $(\mathbb{3}b)c = b(b(bc))$. Очевидно, эта операция с таким же успехом может быть применена к любому другому натуральному числу Черча. (В действительности, операция $\lambda abc.[(ab)(bc)]$ приводит к тому же результату, что и S.)

А как насчет удвоения числа? Удвоение числа может быть получено с помощью операции

$$D = \lambda abc.[(ab)((ab)c)],$$

что легко видеть на примере ее действия на $\mathbb{3}$:

$$\begin{aligned} D = \lambda abc.[(ab)((ab)c)]\mathbb{3} = \lambda bc.[(\mathbb{3}b)((\mathbb{3}b)c)] = \\ = \lambda bc.[(\mathbb{3}b)(b(b(bc)))] = \\ = \lambda bc.[b(b(b(b(b(bc)))))] = \mathbb{6}. \end{aligned}$$

Фактически, основные арифметические операции – сложение, умножение и возведение в степень – могут быть определены, соответственно, следующим образом:

$$\begin{aligned} A &= \lambda fgxy.[((fx)(gx))y], \\ M &= \lambda fgx.[f(gx)], \\ P &= \lambda fg.[fg]. \end{aligned}$$

Читатель может самостоятельно убедиться (или же принять на веру), что

$$\begin{aligned} (A\mathbb{m})\mathbb{n} &= \mathbb{m} + \mathbb{n}, \\ (M\mathbb{m})\mathbb{n} &= \mathbb{m} \times \mathbb{n}, \\ (P\mathbb{m})\mathbb{n} &= \mathbb{n}^{\mathbb{m}}, \end{aligned}$$

где $\mathbb{m}$ и $\mathbb{n}$ – функции Черча для двух натуральных чисел, $\mathbb{m} + \mathbb{n}$ – функция, выражающая их сумму, и т.д. Последняя из этих функций поражает больше всего. Посмотрим, например, что она дает в случае $\mathbb{m} = \mathbb{2}$, $\mathbb{n} = \mathbb{3}$:

$$\begin{aligned} (P\mathbb{2})\mathbb{3} = ((\lambda fg.[fg])\mathbb{2})\mathbb{3} = (\lambda g.[\mathbb{2}g)]\mathbb{3} = \\ = (\lambda g.[\lambda fx.[f(fx)]g])\mathbb{3} = \lambda gx.[g(gx)]\mathbb{3} = \\ = \lambda x.[\mathbb{3}(\mathbb{3}x)] = \lambda x.[\lambda fy.[f(f(fy))](\mathbb{3}x)] = \\ = \lambda xy.[(\mathbb{3}x)((\mathbb{3}x)((\mathbb{3}x)y))] = \\ = \lambda xy.[(\mathbb{3}x)((\mathbb{3}x)(x(x(xy))))] = \\ = \lambda xy.[(\mathbb{3}x)(x(x(x(x(xy))))))] = \\ = \lambda xy.[x(x(x(x(x(x(x(x(xy))))))))] = \mathbb{9} = \mathbb{3}^{\mathbb{2}}. \end{aligned}$$

Операции вычитания и деления определяются не так легко (на самом деле нам потребуется соглашение о том, что делать с ($\mathbb{m} - \mathbb{n}$), когда $\mathbb{m}$ меньше $\mathbb{n}$, и с ($\mathbb{m} / \mathbb{n}$), когда $\mathbb{m}$ не делится на $\mathbb{n}$). Решающий шаг в развитии этого метода был сделан в начале 1930-х годов, когда Клини удалось найти выражение для операции вычитания в рамках схемы Черча! Затем были описаны и другие операции. Наконец, в 1937 году Черч и Тьюринг независимо друг от друга показали, что всякая вычислимая (или алгоритмическая) операция – теперь уже в смысле машин Тьюринга – может быть получена в терминах одного из выражений Черча (и наоборот).

Это воистину замечательный факт, который подчеркивает глубоко объективный и математичный характер понятия вычислимости. На первый взгляд, понятие вычислимости по Черчу не связано с вычислительными машинами. И тем не менее, оно имеет непосредственное отношение к практическим аспектам вычислений. В частности, мощный и гибкий язык программирования LISP включает в себя как существенный элемент основные структуры исчисления Черча.

Как я отмечал ранее, существуют и другие способы определения понятия вычислимости. Несколько позже, но независимо от Тьюринга, Пост предложил во многом сходную концепцию вычислительной машины. Тогда же благодаря работам Дж. Хербранда и Гёделя появилось и более практичное определение вычислимости (рекурсивности). Х.Б. Карри в 1929 году, и ранее, в 1924, М. Шенфинкель, предложили иной подход, который был отчасти использован Черчем при создании своего исчисления (см. Ганди [1988]). Современные подходы к проблеме вычислимости (такие как машина с неограниченным регистром, описанная Катлендом [1980]) в деталях значительно отличаются от разработанного Тьюрингом и более пригодны для практического использования. Однако понятие вычислимости во всех этих подходах остается неизменным.

Как и многие другие математические идеи, особенно наиболее фундаментальные и красивые, идея вычислимости кажется овеществленной и объективно существующей в платоновском смысле. Именно к этому мистическому вопросу о платоновской реальности математических понятий в целом мы и обратимся в следующих двух главах.

(Продолжение в файле {PENRO2})

* * *

2010.11.24 01:42 ночь на среду

В.Э.: Пенроуз в конце §2.2 написал: «*А не являются ли интуитивные прозрения сами алгоритмическими? Это один из вопросов, которые будут для нас важны в дальнейшем.*»

Разумеется, являются! В мозге-компьютере нет ничего, кроме программ, работающих, конечно же, по тому или иному алгоритму. Здесь я сделаю еще одну – третью – вставку с моим ответом профессору Тамбергу, написанным более десяти лет назад:

(...)

§25. Интуиция и «иррациональное» мышление

2000.05.18 15:22 четверг
(раньше на 10 лет, 6 месяцев, 5 дней, 10 часов, 20 минут)

.284. Вы пишете:

> .285. «В своей крайней последовательности научный метод приводит к натуралистическому мировоззрению, согласно которому наука в состоянии дать ответы на абсолютно все вопросы нашей жизни. Человек с этой точки зрения представляется таким же познаваемым объектом природы, как и все остальные. Он не является ничем особым и «специальным».[145] Однако научное творчество и открытия возможны лишь интуитивным путем, где они проявляются как вспышки новых идей или решений проблем, непостижимые умом и имеющие иррациональную основу.[146]» (с. 209–210).

.286. Вся использованная здесь система понятий (такие понятия как «интуитивный», «рациональный», «иррациональный» и т.д.) полностью игнорирует всякое представление о том, как эти вещи можно было бы реализовать в биологическом (или промышленном) компьютере. Посмотрим все-таки, как это выглядит тогда, если такое представление у человека имеется и пускается в ход.

.287. Итак, по представлениям (по модели), используемым в Вашей цитате, существуют два способа мышления: 1) рациональный или логический; 2) иррациональный или интуитивный. И для творчества главным является второй.

.288. Я лишь в небольшой степени могу повторить здесь то, что уже многократно говорилось во многих местах моих сочинений о принципиальном устройстве человеческой операционной системы, поэтому мне приходится полагаться на то, что всё это в общих чертах уже известно (см., напр., LASE2-50.стр. и окружающее {SKATI.605}).

---

[145] Klīve Visvaldis. «Pa kuru ceļu? Pārdomas par iespējamām atbildēm uz mūsu dzīves lielajiem jautājumiem». Lincoln, Nebraska: LELBA apgāds, 1988. (Кливе Висвалдис. «По которому пути? Размышления над возможными ответами на большие вопросы нашей жизни». Линкольн, Небраска (США): LELBA, 1988).

[146] Фейнберг Е.Л. «Две культуры: Интуиция и логика в искусстве и науке». Наука, Москва, 1992 (на русском языке).

.289. Итак, человек (его мозг) должен решить какое-то задание *З*. Здесь не существенно, что это за задача: доказательство ли теоремы Пифагора, или решение какого-то уравнения, или создание какой-то новой концепции и т.д. (или удостовериться в существовании Бога...).

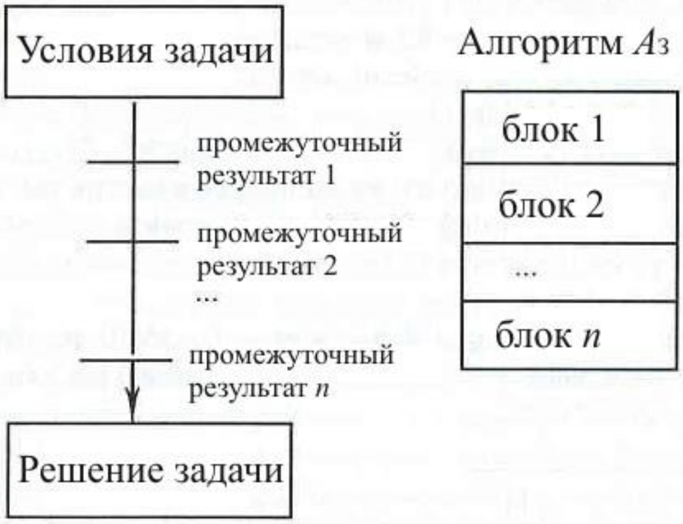
.290. Если мозг человека представляет собой биологический компьютер (что, как известно, является постулатом одной модели), то задачу *З* этот компьютер решает по какому-то алгоритму $A_З$ (это не важно, откуда появляется этот алгоритм $A_З$, – его создание является для мозга просто немножко более ранней задачей, к которой относится всё то же самое, что мы сейчас решим о задаче *З*).

.291. Тогда принципиальная схема решения задачи *З* такова:

.292.

.293. Более детализировано этот же процесс мы можем изобразить так:

.294.



.295. В общем случае путь решения задачи состоит из каких-то *n* звеньев, которым отвечают соответствующие шаги или блоки алгоритма, каждый из которых дает какие-то промежуточные результаты и т.д., – следовательно, существует то, что мы здесь ниже будем называть ходом решения задачи.

.296. Допустим, что компьютер (биологический или промышленный) отработал по приведенной выше схеме и получил решение задачи *З*. Каким это решение является: рациональным или иррациональным? логическим или интуитивным?

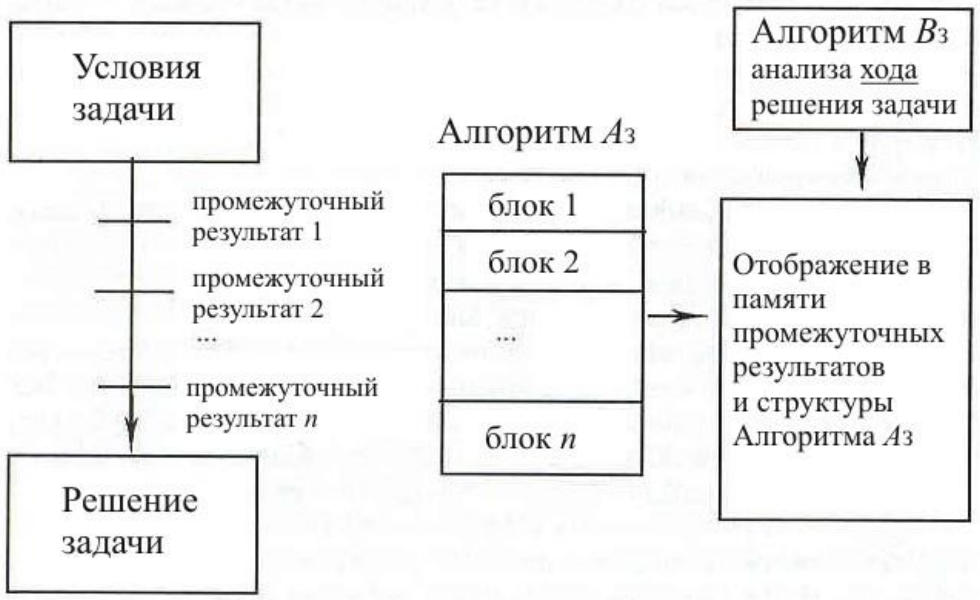
.297. Вы по-видимому скажете, что это рациональное и логическое решение (так как происходит же по алгоритму).

.298. А я отвечу: «Ерунда! Это типичное, классическое иррациональное и интуитивное решение!».

.299. Тогда Вы, естественно, спросите меня: «А какое же в таком случае решение является рациональным и логическим?».

.300. Чтобы решение стало таким, какие мы в «быту» называем «рациональными» и «логическими», компьютер должен быть способным проанализировать ход решения. А в приведенной выше схеме не было никаких элементов, которые это делали бы или этому способствовали бы. Поэтому дополним предыдущую схему указанными элементами:

.301.



.302. Вот теперь другое дело! Теперь в памяти компьютера (биологического или промышленного) имеются сведения о структуре алгоритма *А*3, о его работе и промежуточных результатах; он «помнит» и «знает», что и как делал, и он имеет также алгоритм *В*3, который эту деятельность может оценить (как правильную или неправильную и т.д.). Теперь решение задачи *З* стало «логическим» и «рациональным» (конечно, если только алгоритм *В*3 признает Ход решения правильным; если нет, то имело место не решение, а «просто ошибка»).

.303. Следовательно, как мы видим, «интуитивным» решением является такое, при котором результат получен, а каким способом получен, каким путем найден, – об этом никаких сведений в компьютере нет (и, значит, Ход решения не может быть проверен и всесторонне проконтролирован по многим критериям).

.304. Не надо быть специалистом по информатике, чтобы увидеть, что первое («интуитивное») решение намного проще, чем второе. В первом случае надо только получить алгоритм *А*3, – и всё. Во втором случае надо получить этот же алгоритм, плюс еще весь ход решения фиксировать в памяти, плюс еще получить алгоритм *В*3 со всеми встроенными в него критериями оценки Хода решения...

.305. Поэтому, конечно, не удивительно, что почти все «новые идеи» первоначально появляются «интуитивным» (т.е. более простым) путем, и только потом осуществляется вся остальная работа, чтобы довести дело до совершенного состояния.

.306. Так это выглядит, если человек МОЖЕТ себе представить, каким образом всё это реализовать в компьютере. Как видите, вся «таинственность» и «сверхъестественность» с «интуиции» сразу спадает; это никакая не «высшая» форма мышления, а совсем наоборот, простейшая и первоначальная. Подавляющее большинство промышленных компьютеров «мыслят» именно интуитивно – т.е., просто руководствуясь данной программой, не запоминая ход решения задачи и не анализируя «законность» и «правильность» этого хода решения. Чтобы организовать такую проверку, необходима (очень большая) дополнительная работа (которую осуществляет – если вообще осуществляет – только человек, «программист», не возлагая это на промышленный компьютер).

.307. Отсюда и общая оценка «интуитивному мышлению»: конечно, оно полезно и нужно (мозговому компьютеру) как самый простой и быстрый способ получения решения, но в то же время он не может (не должен) оставаться единственным и окончательным. Хорошо, если алгоритм *А*3 давал правильное решение задачи *З*; а если давал неправильное или неточное – то (при интуитивном мышлении) оно так и останется как единственное решение: без проверки, без контроля, без постепенного улучшения. Поэтому там, где за первоначальным «интуитивным» и «иррациональным» решением не следует дальнейшая «логическая» и «рациональная» работа, – там общие результаты весьма печальны.

.308. «*Попросту говоря,* – Вы пишете[147] – *отличие между научным и религиозным методом познания истины заключается в пропорциях – наука преимущественно опирается на логически рациональный разум человека, но не может обойтись без иррационального (интуитивного) момента, а религия, напротив, основывается на божественном Откровении, т.е. на трансцендентальном моменте*» (стр.210).

.309. С точки зрения рассматриваемой здесь модели (Веданской), «Откровение» представляет собой типичное интуитивное решение какой-то задачи: мозговой компьютер «просто решает», что «это так»; мотивы решения, Ход решения не фиксируются и не анализируются. Потому-то в этом случае и бывает так легко отбросить принцип Оккама (который был бы типичной составной частью алгоритма B_3) и подобные критерии оценки.

*

Так я тогда ответил профессору Тамбергу. Сказанное ему – пригодится в дальнейшем и нам. Пенроуз под «интуицией» понимает не только это, но и некоторые другие вещи – в частности: знания о работе и устройстве мозговых программ (не отдавая, разумеется, себе отчета в том, что речь идет именно о программах).

* * *

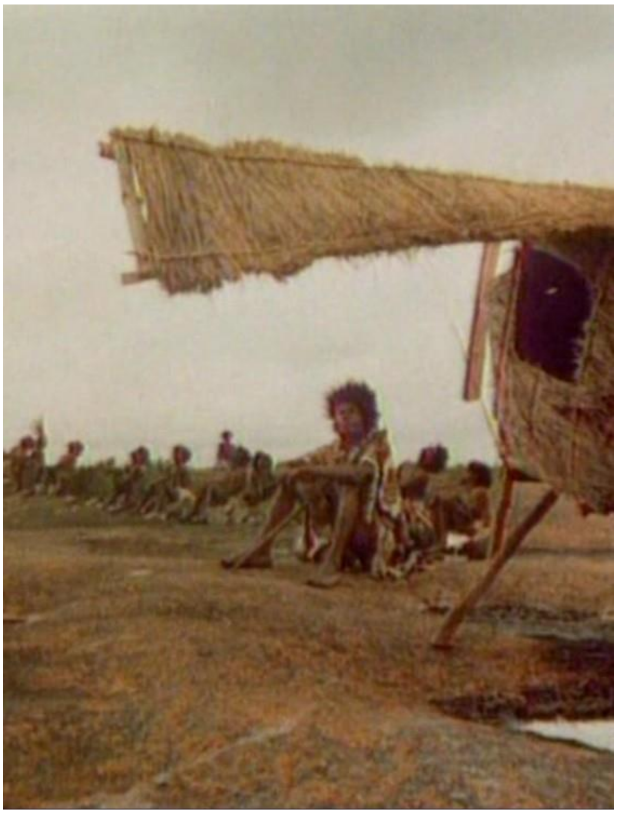
(Конец вставки)

---

[147] **P.S.** Покойный профессор Юрис Тамберг (МОИ № 53, стр.32) был интересной, но с нашей точки зрения довольно противоречивой личностью. Будучи хабилитированным доктором ф.-м. наук, он долгие годы был сотрудником Саласпилсского атомного реактора, а после его закрытия – сотрудником Института физики твердого тела Латвийского университета. Однако лекции он читал не только на Физико-математическом факультете Университета, но и на Теологическом факультете, и был одним из основателей и активистов «Группы диалога науки и религии». Разбираемая мною его статья была опубликована в альманахе, издаваемом именно Теологическим факультетом. В то время как «чистые ученые», никак не связанные с религией, начисто отказывались как-либо рассматривать Веданскую теорию, профессор Тамберг дал о ней положительную рецензию, единственную в ее истории. Однако эта «положительность» рецензии заключалась в утверждении того, что Веданскую теорию нужно всесторонне изучить и оценить, а не в том, что она верна. Профессор Тамберг не подтвердил правильность теории и не принял ее как свое собственное убеждение, а только пожелал мне успехов и достижений. Его позицию, может быть, лучше всего характеризуют его слова, одни из самых первых, что он мне сказал, когда мы впервые встретились: «Я верю в Бога, но это ничего!»

Файл PENRO2

http://vekordija.narod.ru/R-PENRO2.PDF



The Emperor's New Mind

Большинство математиков предало забвению древние традиции математики и наследие ее прошлого. Наполненные глубоким содержанием сигналы, которые посылает нам природа, достигает лишь закрытых глаз и нечутко прислушивающихся ушей. Математики продолжают жить на проценты от репутации, заработанной их предшественниками, и жаждут при этом шумного одобрения и такой же поддержки, какую математика имела в прошлом.

Морис Клайн,[148] 1980

Предисловие в Векордии

2010.11.11 10:26 четверг

В этот том Векордии помещаются две главы Первой книги Пенроуза, посвященные непосредственно математике, – и тем самым наиболее тесным образом касающиеся самого главного предмета Веданской теории. Сейчас, когда я пишу это Предисловие, я еще не приступил к комментированию этих глав, но чувствуется, что в этом томе, по-видимому, моих комментариев окажется так много, как ни в одном другом из девяти томов, представляющих идеи Роджера Пенроуза в настоящем моем Дневнике.

Поэтому представляется полезным напомнить читателю хотя бы главные моменты из того основания, на которое эти комментарии опираются. Здесь нет ни возможности, ни цели излагать Веданскую теорию в сколь-нибудь широком виде, но попытаюсь повторить самое основное из

[148] Клайн Морис. «Математика: утрата определенности». «Мир», Москва, 1984, с 351.

того, что многократно говорилось мною на протяжении более 32 лет в разных моих сочинениях и комментариях к чужим сочинениям, – чтобы, возможно, помочь читателю лучше понять то, что говорится ниже в этом томе.

1. Начало математики

Всегда, когда Вы изучаете какой-нибудь предмет, небесполезно бывает знать, ЧТО, собственно, Вами изучается, ЧТО этот предмет из себя представляет. Когда Вы изучаете «то, не знаю что», то легко сбиться и уплыть во всякие фантазии, не соответствующие никакой действительности.

Это относится и к математике. Если кто-то (как Пенроуз и тысячи других математиков и философов) хочет рассуждать о математике, то будет весьма полезно в первую очередь установить, что же такое математика и что это за вещи, которые этой наукой изучаются.

Разумеется, Пенроуз (и тысячи других математиков и философов) скажут, что они это знают. (Правда, говорят они это как-то не очень уверенно). Однако их объяснения этих вещей нельзя признать ни в малейшей степени удовлетворительными. (Я хотел здесь привести пример и с этой целью снова заглянул в некоторые энциклопедии, но там всё настолько расплывчато и обширно, что примеры загромоздили бы мне всё Предисловие, ничего существенного так и не сказав. Так что загляните лучше сами в какую угодно из «официальных» книг: попытайтесь по ним уяснить, что такое математика, что такое число и т.д.).

А чтобы действительно уяснить, что такое математика и что из себя представляют те объекты, которые ею изучаются (в первую очередь числа), нужно вернуться к ее истокам и посмотреть, с чего и в каких условиях она начиналась, а потом немножко подумать. Итак, как же математика начиналась?

Если мы придерживаемся не каких-нибудь религиозных, а научных взглядов, то картина выглядит так. Ходил по Земле первобытный человек; он представлял из себя материальную систему, состоящую из атомов и молекул, и в этой системе работали различные подсистемы, выполняющие различную работу: сердце разгоняло по организму кровь, кровь подносила клеткам пищу и кислород – и т.д. А одна подсистема занималась обработкой информации о внешнем мире и о самом организме. Вот эта-то подсистема и создала в конце концов математику.

Но КАК система обработки информации может создать математику? (Или первоначально – просто числа).

Что она для этого должна делать? КАК она для этого должна работать?

Чтобы ответить на эти вопросы, требуется, конечно, некоторое понимание вообще о принципах работы систем обработки информации – т.е. о компьютерах.

2. Классификация множеств программой N

Когда такое понимание есть, то становится видным и ответ: для создания натуральных чисел система в первую очередь должна уметь классифицировать объекты (внешнего мира) по количеству элементов в них. То есть, должна уметь различить, когда элементов три, а когда четыре и т.д., и относить объекты с тремя элементами к одному классу (множеству), а объекты с четырьмя элементами к другому и т.д.

Но системы обработки информации (т.е. компьютеры) «умеют» что-то делать лишь тогда, когда у них имеется для этого программа. Следовательно, чтобы первобытные люди могли начинать классифицировать объекты по количеству элементов в них, в их мозге должна была уже существовать программа такой классификации. (А создать эту программу они должны были, как и все другие программы своей деятельности, путем самопрограммирования).

Таким образом, эта мозговая программа классификации объектов по количеству элементов (назовем ее программой N)[149] становится самым первым объектом у истоков математики. Математика начинается с программы N.

Но как же работает эта программа N? Как она осуществляет свою классификацию? Перебрасывает все объекты внешнего мира, имеющие три элемента, в одну кучу, а все объекты, имеющие четыре элемента, в другую кучу? Разумеется, нет – во внешнем мире она ничего не трогает и ничего не перемещает. Она просто создает внутримозговую (внутрикомпьютерную) структуру («таблицу»), которую мы могли бы назвать «множеством всех объектов с тремя элементами», и потом устанавливает связку между данным конкретным объектом (точнее: его

[149] От латинского *naturalis* – естественный или от *numerus* – количество, член, часть.

внутрикомпьютерным представлением) и этой «таблицей». Такая связка (для всей дальнейшей работы мозгового компьютера) будет означать: «этот объект принадлежит множеству всех объектов с тремя элементами».

Так появляются знакомые нам из математики вещи, но только теперь мы имеем неизмеримо более точное представление об истинной природе этих вещей. Математики – вроде Пенроуза – могут сколько угодно рассуждать об «абстрактных множествах», да только всё то, что они о тех говорят и знают, не дает им возможности самим построить существо, размышляющее о математике (и Пенроуз даже думает, что такое существо построить вообще невозможно на основе программ). А то, что только что сказал я, может в компьютере осуществить любой мало-мальски квалифицированный программист, и, продолжая в таком же духе, он построит существо, реализующее математическое мышление. Поэтому-то наше (изложенное здесь) представление о числах и множествах и является несравнимо более глубоким, нежели у математиков. Нельзя ставить на одну полку знания человека, который телевизор построить не может, и человека, который это сделать может. И нельзя ставить на одну полку знания людей, которые математическое мышление сами создать не могут, и тех, кто это может.

Рис. VE1. Исходная концепция множества в Веданской теории. Рассуждая о всяком множестве, нужно различать три главные связанные с ним вещи: 1) Реалию множества, существующую вне ментального мира; 2) Номиналию множества, существующую в ментальном мире и соответствующую реалии; 3) Мозговую программу, способную отличить, принадлежит ли объект данному множеству или нет.

3. Конкретные множества

Итак, «множество всех объектов с тремя элементами» в компьютере (мозге) представлено внутрикомпьютерным объектом – «таблицей». Такая «таблица» в Веданской теории называется номиналией. (Нужно же присваивать какие-то названия тем вещам, о которых мы говорим, иначе невозможно ничего сказать). Номиналия – это внутримозговой (внутрикомпьютерный) объект, соответствующий внешнему объекту (называемому реалией).

Когда реалия представляет собой просто один определенный объект внешнего мира (например, «вот эти три яблока, лежащие сейчас на столе»), то ситуация довольно проста. Во внешнем мире три яблока; внутри мозга – их номиналия (как мозговая структура).

Немножко сложнее дело обстоит, когда реалия представляет собой такой не совсем четко очерченный объект как «множество всех объектов с тремя элементами». Ведь на самом деле мозг не может действительно перебрать все такие объекты! Но он и не перебирает. На самом деле в мозге имеется только номиналия (якобы соответствующая «множеству всех объектов с тремя элементами») и программа N, способная определить, принадлежит ли данный конкретный объект этому множеству, или нет (и устанавливающая связь между номиналией множества и номиналией объекта, если принадлежит).

Это общий принцип: все математические множества в мозге (т.е. в компьютере)[150] представлены двумя главными объектами: 1) номиналией этого множества и 2) мозговой программой, которая способна определить, принадлежит ли рассматриваемый объект этому множеству, или нет. Поэтому, когда мы хотим рассуждать о множествах точно (точнее, чем это делают математики), то мы должны от «абстрактных» (т.е. расплывчатых) рассуждений о «множествах вообще» перейти к рассуждениям о номиналиях и мозговых программах этих множеств.

4. Абстрактные множества

Но вернемся к программе N, с которой начиналась математика. Итак, эта программа классифицирует объекты по количеству элементов в них. Как результат этой классификации она создает целый ряд множеств: «множество всех объектов с одним элементом», «множество всех объектов с двумя элементами», «множество всех объектов с тремя элементами» и т.д. (т.е., на самом деле она создает в мозге номиналии этих множеств, и сама же и способна определить, к которому из этих множеств относится поданный ей на вход конкретный объект).

Ясно, что тут где-то «истоки понятия числа». Но где оно – собственно число?

Пенроуз ниже в этом томе расскажет, что Фреге считал числом «3» собственно «множество всех объектов с тремя элементами» (т.е. в наших терминах – реалию этого множества). Сам Пенроуз в этом вопросе более осторожен: ведь он понимает, что Вселенная может оказаться конечной, и тогда множество чисел не будет бесконечным. Поэтому он добавляет, что понятие числа, правда, начинается с этих множеств Фреге, но потом превращается в некую «абстракцию» или «расширение» (но при этом, разумеется, Пенроуз не может сказать, как же нам это «абстрагирование» встроить в компьютер математического мышления, поэтому его объяснение просто заменяет одно не до конца понятное слово другим не до конца понятным словом).

Всё, что мы до сих пор рассмотрели в связи с программой N, очевидно, «стоит близко» к «истокам чисел», но пока что ни один из введенных нами объектов не может быть собственно числом. Числом не могут быть номиналии (потому что мозговая программа N данного конкретного индивида за всю жизнь построит сравнительно немного номиналий – номиналии лишь тех множеств, которые этот индивид действительно пересчитал; кроме того, тогда у каждого индивида вообще имеются «свои числа»). Числами не могут быть и реалии этих множеств (потому что даже в самом крайнем случае – если пересчитать всю Вселенную – это всего лишь подход Фреге).

Тогда где же собственно числа – объекты, обладающие всеми свойствами чисел, и в первую очередь – их бесконечным количеством? И одинаковой «надежностью» и «реальностью» всех чисел – что простой единицы 1, что числа $1000^{1000^{1000^{1000^{1000}}}}$, безнадежно далеко выходящего «за пределы Вселенной».

И вот тут нам нужно вспомнить об одной вещи, которую, как показывает опыт 32-летней истории Веданской теории, почему-то почти никто не может понять. Этой вещью является анализ программы без ее выполнения.

Вообще программисты каждый свой рабочий день выполняют это дело. Программист сидит над листингом (распечаткой) своей программы и «в голове» «прокручивает» ее: что будет, если ее запустить на выполнение? что будет, если ей подать такие данные? а что, если такие? а что произойдет в таком случае? а что в таких условиях?

Но мало кто (даже среди программистов) понимает, что эту работу может выполнить и компьютер. Эта же программа (A) может сидеть в компьютере (но не выполняться!), и можно

---

[150] При описании Веданской теории воспользуемся введенным Пенроузом делением интересующих нас «сущностей» на три мира: Физический мир, Ментальный мир и Платоновский мир {PENRS4}, которое очень удобно для наших целей. Тогда всё, что находится внутри мозга-компьютера, представляет собой Ментальный мир, а всё, что вне его – либо Физический, либо Платоновский мир.

запустить другую программу (B), которая с этой первой программой (A) будет проделывать в принципе ту же работу, что и сидящий над листингом программист. (Что для этого требуется, как программы должны быть структурированы, – это уже другой вопрос, технический).

Вот эта вторая программа (B), анализирующая первую программу (A) отобразит результаты своей работы в некоторых внутрикомпьютерных структурах (кто понимает программирование, тот представит себе), опять же в «таблицах». Эти «таблицы» (построенные программой B) будут представлять, т.е. кодировать потенциальные результаты (продукты) выполнения программы A. Программа A не работала, настоящих своих продуктов не создавала. Но отработала программа B и создала структуры (назовем их тоже номиналиями), кодирующие то, что программа A могла бы создать.

Такие номиналии тоже вроде соответствуют своим реалиям. С точки зрения компьютера и его программ всё выглядит как прежде: есть номиналия внутри компьютера, ей соответствует реалия вне компьютера – всё в порядке. Но только теперь реалией такой номиналии является не материальный объект физического мира, а «идеальный» объект «платоновского мира» (впрочем, совсем не произвольный, а с точки зрения компьютера – мозга – столь же реальный и объективный, как и прежние объекты; его свойства не взяты «с потолка», а определяются программой A).

Такой анализ одной программы, осуществляемый другой программой, – обычная работа в человеческом мозге (и составляет существенную компоненту интеллекта). Ведь живым существам нельзя любую программу, составленную их аппаратом самопрограммирования, сразу пускать на выполнение. Нужно сначала «подумать», т.е. проанализировать возможные последствия ее выполнения. Так что этот аппарат создан (естественным отбором) отнюдь не для математики – но он играет огромную роль в сотворении математики.

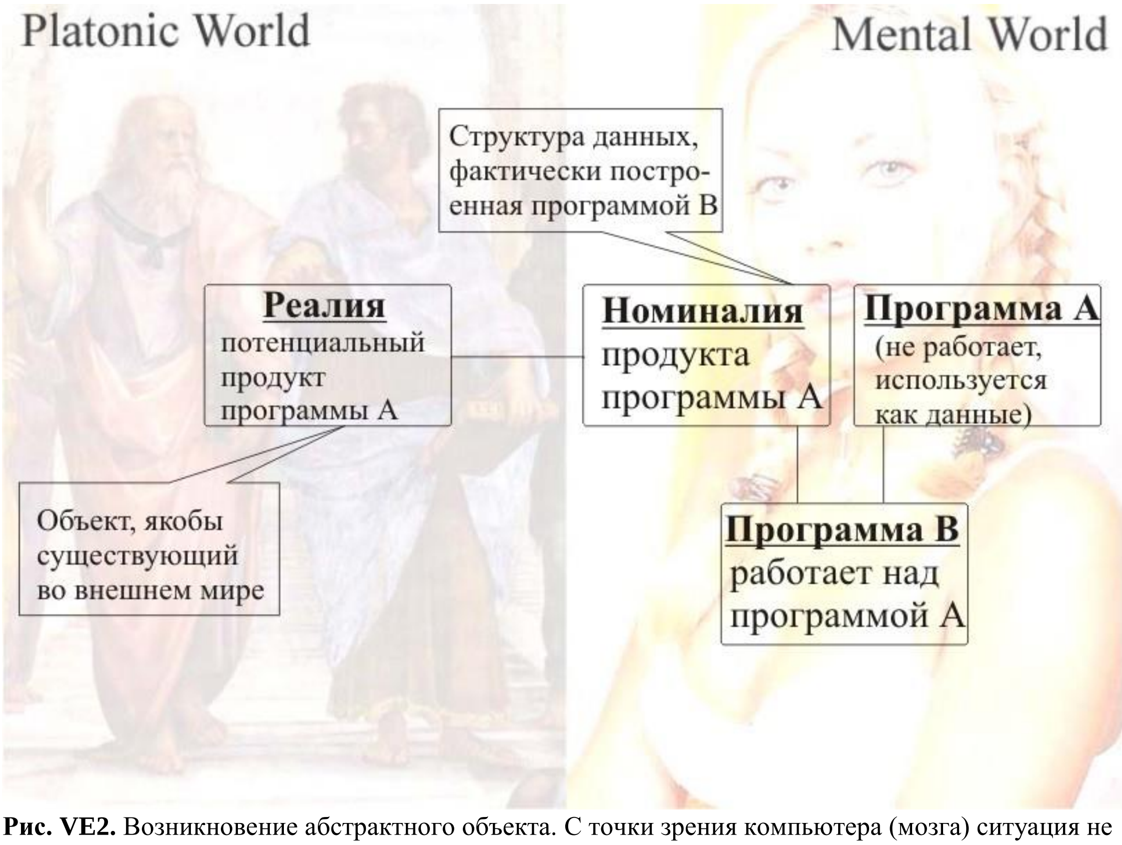


Рис. VE2. Возникновение абстрактного объекта. С точки зрения компьютера (мозга) ситуация не отличается от ситуации на Рис. VE1: номиналия является такой же структурой данных, и ей соответствует реалия «во внешнем мире»

Практически все «абстрактные» объекты математики являются вот такими вот потенциальными продуктами различных (мозговых) программ. Имеется программа (A); ее (не выполняя) анализируют «сбоку»[151] другой программой (В); та строит номиналии потенциальных продуктов программы А в своем компьютере, а реалии этих продуктов «существуют» в «платоновском мире идей».

В отличие от реальных продуктов программы А (когда она действительно отработала), потенциальные ее продукты могут быть бесконечными. Нет проблем! Программа В посмотрела и установила, что А будет работать бесконечно, завела номиналию, и этой номиналии (в платоновском мире) соответствует бесконечное множество.

5. Классификация отношений множеств программой R

2011.01.08 17:05 суббота

Теперь пусть программа В отработает над программой N. В результате будет «создана» реалия, представляющая собой бесконечный ряд таксонов классификации множеств по количеству элементов. В отличие от «множеств Фреге», этой реалии безразлично, конечна или бесконечна Вселенная: – ряд таксонов, потенциальных продуктов программы N, всё равно бесконечен, и все таксоны одинаково реальны – что в начале ряда, что на любом удалении от начала. И они одинаковы для всех людей (и вообще любых субъектов).

Вот это и есть первый объект у нас, обладающий всеми свойствами чисел.

Этот абстрактный объект Платоновского мира мы можем считать натуральными числами.

Легко видеть, каким образом в этом множестве вводятся арифметические операции. Они представляют собой объективно существующие соотношения между таксонами. Так, если одно множество программой N квалифицируется как принадлежащее таксону 3, а второе как принадлежащее таксону 4, то их объединение обязательно будет классифицироваться как принадлежащее таксону 7 – и т.д.

2011.01.13 14:38 четверг

Однако программа N, классифицирующая множества, хоть и была способна положить начало математике путем создания самых первых чисел, не может обеспечить сколь-нибудь далекое ее развитие. При использовании этой программы субъект имеет только натуральные числа – и никаких больше: ни дробных, ни отрицательных, ни иррациональных, ни комплексных...

Но реальные ситуации и решаемые в них задачи вскоре вынуждают субъекта переходить от классификации множеств (программа N) к классификации отношений между множествами (такая программа классификации в Веданской теории называется «программой R»[152]). Отношения между множествами в исходных данных программы R представлены в виде пар множеств. В каждой паре одно множество считается «единицей измерения», а другое множество – «измеряемое». В один таксон классификации попадают те пары множеств, в которых соотношения между единицей и измеряемым «одинаковы» (по критериям программы R).

Как и в случае с программой N, программа R тоже создает бесконечные множества таксонов-чисел в платоновском мире по схеме рисунка VE2.

Одно подмножество таксонов-чисел программы R оказывается во всех отношениях эквивалентным (изоморфным) продукции программы N; это те соотношения между множествами, когда единица измерения входит в измеряемое «целое число раз». Этим подмножеством программа R как будто «переопределяет» натуральные числа, созданные ранее программой N. Отныне субъекту становится безразлично, понимать ли под натуральными числами таксоны, созданные программой N (обозначим множество этих таксонов как N_N), или считать натуральными числами упомянутое подмножество таксонов программы R (обозначим их N_R). Числа N_N и N_R для субъекта сливаются вместе, начиная уже затуманивать понятие числа и понемножку создавать вековую проблему «Что же такое число?!».

Но остальные таксоны программы R тем временем порождают дробные числа (как соотношения множеств, например, с количеством элементов 7 и 3, и т.п.). Пока программа R

[151] Из-за этой ассоциации о том, что обе программы находятся рядом, «соприкасаются боками», и при этом одна из них анализирует, изучает другую, – такой процесс в Веданской теории называется бокоанализом.

[152] От латинского *relatio* – отношение.

применяется только к конкретным множествам[153] (точнее: к их номиналиям), эти дробные числа остаются «рациональными». Но когда программу R применяют к соотношениям абстрактных множеств (таким, например, как «соотношение окружности и диаметра» или «соотношение стороны квадрата и его диагонали»), то дроби могут стать «иррациональными».

Пока программа R интересуется только количеством элементов в множествах, составляющих измеряемые пары (соотношения), создаются только те множества чисел, которые в традиционной математике называются «положительными числами», а в Веданской теории «метрическими».

Но в общем случае программа R учитывает не только количество элементов в множествах данной пары, но и взаимное расположение обоих множеств.

Интерес представляют две модификации программы R:

R′, учитывающая линейную ориентацию множеств пары: одинаково или противоположно они направлены; программа R′ порождает отрицательные числа классической математики (а с точки зрения Веданской теории: она разбивает метрические числа на положительные и отрицательные);

R″, учитывающая планарную ориентацию множеств пары; она порождает «комплексные» числа традиционной математики.

6. Первичные и вторичные алгоритмы

Так, развивая у себя в голове мозговые программы классификации множеств по количеству элементов и пар множеств по их отношениям, первобытный человек создал «числовые системы» и положил начало математике. (Вторым аналогичным направлением его деятельности было развитие мозговых программ «конструирования»: проведения линий, рисования фигур и т.п., и изучение потенциальных продуктов таких программ. Легко понять, что речь идет о геометрии; там сохраняются все те же закономерности: программы, их продукты, бокоанализ программы А программой В и создание ею объектов Платоновского мира; номиналии, реалии и т.д. Но в этом Предисловии, чтобы не загромождать рассказ, не будем углубляться в геометрию, поясняя все нужные нам понятия Веданской теории только на примере чисел).

Итак, числа представляют собой объекты Платоновского мира (причем «платоновский мир» понимается в смысле рисунка VE2). Числа являются таксонами классификаций множеств или их отношений по программам N, R, R′, R″. Только эти объекты обладают всеми свойствами чисел, известными нам со школьной скамьи, такими как их бесконечность, одинаковая «реальность» в любом месте «числовой оси», одинаковость для всех субъектов и «абстрактность», т.е. существование «вне пространства и времени».

В мозгах (компьютерах) субъектов существуют программы работы непосредственно с числами (как с таксонами классификаций по программам N, R, R′, R″). «Визуализация» операции умножения, приведенная Пенроузом в §1.19 тома {PENRS1}, представляет собой типичный пример работы программ этой группы. Там мозговые программы по-разному измеряют (классифицируют) множества строк, множества столбцов и множества суммарных элементов «визуализируемой» матрицы.

Такие операции непосредственно с множествами и с таксонами-числами в Веданской теории называются первичными. (Соответственно, мозговые программы, эти операции осуществляющие, называются первичными программами).

Однако первичными операциями при помощи первичных программ не многого достигнешь. С них математика только начинается, а своего расцвета и неимоверной мощи она достигает при помощи вторичных операций, осуществляемых, соответственно, вторичными программами мозга.

Вторичные действия над числами начинаются с того, что числа (т.е. таксоны классификации) как-то обозначаются. Такое (графическое) обозначение числа в Веданской теории называется нотатой. Всем нам известны «римские» нотаты: I, II, III, IV... и «арабские» нотаты: 1, 2, 3, 4... (известны и другие нотаты: греческие, славянские, еврейские и т.п.). Впредь ограничимся «арабскими» нотатами в «десятичной системе счисления».

[153] «Конкретным множеством» в Веданской теории называется множество, в номиналии которого перечислены все его элементы. Противоположность: «абстрактное множество», элементы которого не перечисляются конкретно, а (в номиналии) указывается ссылка на программу, которая это множество создает (на программу, потенциальным продуктом которой это множество является).

Сущность вторичных действий состоит в том, что первичные действия (над множествами и числами-таксонами) заменяются действиями над нотатами. Типичный пример: пенроузовское

$$3 \times 5 = 5 \times 3.$$

Те действия, которые только что при «визуализации» проделывались первичными программами, теперь закодированы нотатами (обозначающими числа, «арифметические операции», определенные соотношения – «равенство» и т.п.).

Вот здесь, во вторичных операциях, математика и начинает по-настоящему разворачиваться. Вскоре ей недостаточно уже 3 × 5 = 5 × 3, а надо писать нотаты $a \times b = b \times a$ (а потом и еще похлеще). Но пусть нас не пугают всё более и более хитроумные системы математических нотат и вторичных операций. Просто нужно помнить главное:

1) система вторичных обозначений и действий стартовала с платформы первичных действий; то, что 3 × 5 = 15 вытекает НЕ из самих вторичных нотат и действий с ними, а из того обстоятельства, что в первичных операциях программа классификации N всегда установит принадлежность множества элементов матрицы таксону 15, если множество строк относится к таксону 3, а множество столбцов к таксону 5;

2) вторичная работа (над нотатами) осуществляется другими мозговыми программами, нежели первичная работа (с множествами); для этих других программ существуют таблицы умножения, алгоритмы умножения чисел при помощи сдвинутых строчек, вычитания с «занятием разряда» и т.д.;

3) но вторичная работа с самого начала построена так, чтобы она сохраняла соответствие (изоморфизм) с первичной работой над множествами;

4) именно поэтому и только поэтому вторичная работа (над нотатами) может быть пригодна людям: именно поэтому первичную работу над множествами (их измерение и классификацию) можно заменить вторичной работой над нотатами (вычислениями);

5) такие вторичные построения (над нотатами), которые потеряют соответствие (изоморфизм) с работой первичных программ (над множествами), принципиально не будет иметь никакой возможности применения и пригодности.

Введение в математику вторичных программ, их специфических алгоритмов (вычислительных) и всё более сложных построений над ними исторически еще более замаскировало предмет и сущность математики и окончательно запутало самих математиков. Многие (в том числе Пенроуз) перестали даже различать собственно число и его нотату (его обозначение), готовы считать числом вереницу цифр; они не видят также принципиальных критериев потенциальной пригодности того или иного вторичного построения математики (такого, например, как «теория множеств» Георга Кантора).

7. Термины Веданской теории, применяемые в комментариях

На этом я закончу данное краткое Предисловие. Оно было предназначено для того, чтобы в комментариях к тексту Роджера Пенроуза я мог пользоваться такими понятиями Веданской теории как:

– субъект (человек, животное, робот, инопланетянин и т.п.), обладающий «разумом»,
– номиналия множества в мозге-компьютере субъекта,
– реалия множества во внешнем для мозга-компьютера мире,
– программа (задающая множество),
– программа N (определяющая первичные натуральные числа N_N),
– программа R (определяющая вторичные натуральные числа N_R и дроби),
– программа R′ (определяющая отрицательные числа),
– программа R″ (определяющая комплексные числа),
– метрические отношения (в паре множеств),
– линейная ориентация (в паре множеств),
– планарная ориентация (в паре множеств),
– бокоанализ (программой B программы A),
– потенциальный продукт (программы A, построенный программой B),
– таксон классификации (число),
– конкретные множества (задаются элементами),
– абстрактные множества (задаются программой),
– ментальный мир (обработки информации), существующий в субъекте,
– платоновский мир (потенциальных продуктов программ) вне субъекта,

– нотата (графическое обозначение числа, операции, соотношения и т.п.),
– первичные (алгоритмы, программы) действия над множествами,
– вторичные (алгоритмы, программы) действия над нотатами,

– пользоваться этими терминами, сохраняя при том хоть какую-то надежду на то, что читатель эти слова поймет.

А более частное применение этих понятий – ниже в комментариях к конкретным вопросам, затронутым Пенроузом.

Валдис Эгле

13 января 2011 года

Роджер Пенроуз. «Новый Разум Короля»

(Продолжение; предыдущее в файле {PENRO1})

Глава 3. Математика и действительность

§3.1. Страна Тор'Блед-Нам

Представим себе, что мы совершаем большое путешествие в некий далекий мир. Назовем его Тор'Блед-Нам. Наша телеметрическая система зарегистрировала сигнал, вывела его на монитор и, отфокусировав изображение, мы увидели следующую картину (рис. 3.1):

Что бы это могло быть? Странного вида насекомое? А может быть, темное озеро с многочисленными втекающими в него ручьями? Или огромный причудливой формы внеземной город, с исходящими в разных направлениях дорогами, которые ведут в расположенные поблизости городки и деревушки? Возможно, это остров – и если это так, то давайте поищем поблизости континент, с которым он связан. Для этого «отойдем назад», т.е. уменьшим увеличение наших приборов раз в 15. И вот – посмотрите-ка – этот новый мир предстал перед нашим взором во всей своей полноте (рис. 3.2):

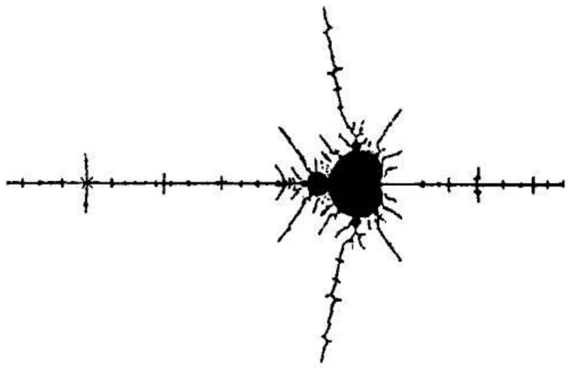


Рис. 3.1. Первый взгляд на новый мир

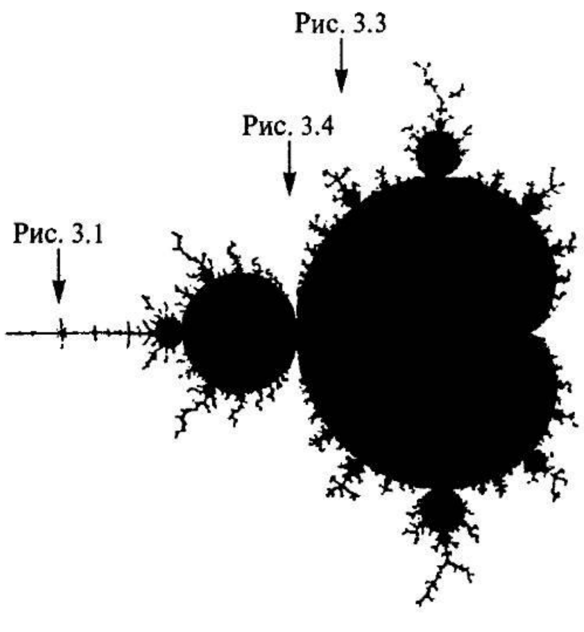


Рис. 3.2. Общий вид Тор'Блед-Нам. Стрелками отмечены области, увеличенные изображения которых даны на рис. 3.1, 3.3 и 3.4

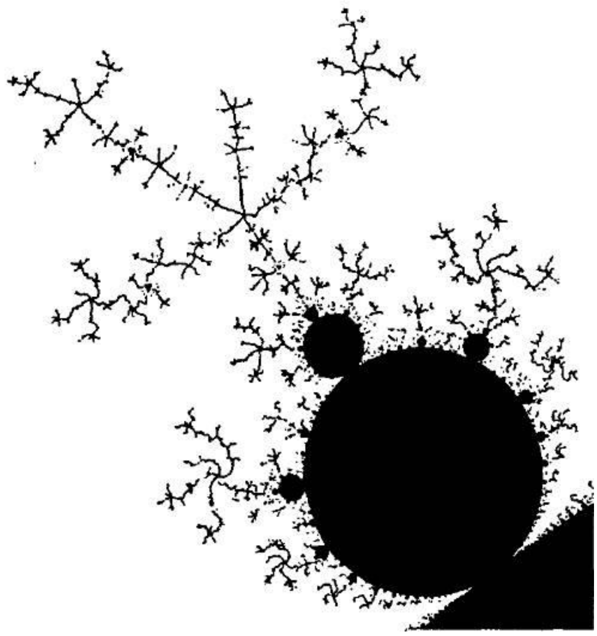


Рис. 3.3. Бородавка с «пятеричностью» своих волоконцев

На рис. 3.2 наш «островок» выглядит как маленькая точка под стрелкой «Рис. 3.1». Все волокна (ручьи, дороги, мосты?), исходящие из первоначального островка, обрываются, за исключением одного – того, что выходит из внутренней части расположенной справа расщелины, и который, в свою очередь, соединен с объектом гораздо большего размера (он изображен на рис. 3.2). Последний, как нетрудно заметить, подобен первоначальному островку, хотя их формы несколько отличаются. При более подробном рассмотрении «береговой линии» выявляются бесчисленные округлые выступы, края которых, в свою очередь, густо усеяны выступами такой же формы. Каждый маленький выступ соединен в каком-нибудь месте с более крупным, и все вместе они образуют бородавчатую структуру, где более крупные выступы покрыты наростами помельче, те – еще более мелкими и т.д. По мере того, как картина становится всё более отчетливой, мы видим мириады мельчайших волокон, исходящих из рассматриваемой структуры. Сами волоконца ветвятся в разных местах, беспорядочно извиваясь. В некоторых частях волокон просматриваются узлы более сложной структуры, неразрешимые при данном увеличении приборов. Ясно, что наш объект – это никакой не остров или континент, и даже не пейзаж. Не исключено, что перед нашим взором чудовищный жук, а то, что мы увидели вначале, – это его детеныш, всё еще соединенный с родителем своеобразной волокнистой пуповиной.

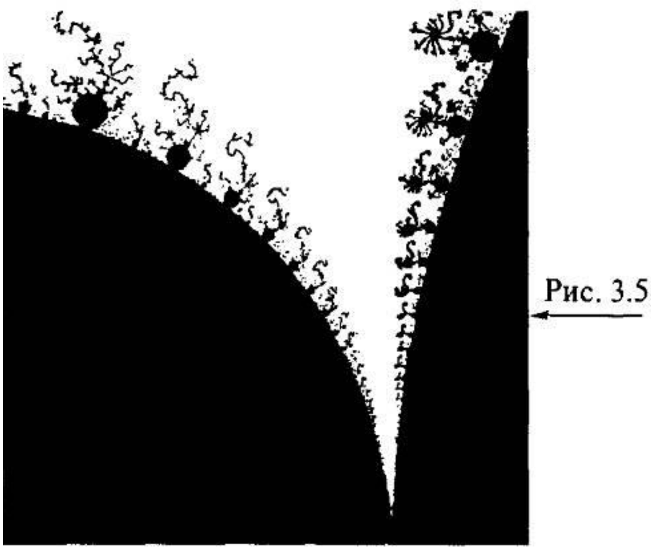


Рис. 3.4. Главная впадина. «Долина морских коньков» едва различима справа внизу

Рис. 3.5. Хвост «морского конька» крупным планом

Давайте исследуем один из наростов у нашего насекомого, для чего увеличим разрешение примерно в десять раз (см. рис. 3.3 – соответствующая область на рис. 3.2. отмечена как «Рис. 3.3»). Своим видом нарост сильно напоминает всё существо целиком, за исключением места соединения. Обратите внимание, что на рис. 3.3 имеется множество точек, в которых сходятся пять волокон. По-видимому, этому конкретному наросту свойственна некая «пятеричность» (точно так же, как для самой верхней «бородавки» на рис. 3.2 характерна определенная «троичность»). На самом деле, если исследовать (на рис. 3.2) расположенный чуть ниже и левее следующий разумного размера нарост, то мы обнаружим у него «семеричность», а у следующего – характерную «девятеричность» и т.д. При углублении во впадину между двумя самыми крупными областями на рис. 3.2, справа будут встречаться наросты с постоянно нарастающим нечетным числом лучей. Давайте всмотримся внимательно вниз вглубь заостренной впадины, повысив увеличение еще в десять раз по сравнению с рис. 3.2 (рис. 3.4). Мы обнаружим множество других мельчайших наростиков на фоне общего беспорядочного завихрения. Справа видны едва различимые спиралевидные структуры, напоминающие «хвосты морских коньков», расположенные в области, которую мы так и назовем – «долина морских коньков». Здесь нам встретятся – если смотреть на это место при достаточно большом увеличении – разнообразные «морские анемоны» или области с богатой флорой. В конце концов, перед нами действительно может быть какой-то экзотический берег – возможно, коралловый риф, изобилующий всевозможными формами жизни. Объект, принятый нами за цветок, при более сильном увеличении может оказаться состоящим из мириада мельчайших и при этом невероятно сложных структур, с многочисленными волокнами и вихреобразными спиралевидными хвостами. Давайте рассмотрим подробнее один из более крупных хвостов морских коньков, а именно – едва различимое образование, обозначенное на рис. 3.4 как «Рис. 3.5» (и соединенное с 29-ричным наростом!). Повысив увеличение в 250 раз, мы увидим изображенную на рис. 3.5 спираль. При этом окажется, что это не обычный хвост: и он тоже состоит из сложнейших вихреобразных структур с многочисленными мельчайшими спиралями и областями в форме осьминогов и морских коньков!

Рис. 3.6. Дальнейшее увеличение места соединения спиралей. В центре едва различим маленький детеныш

Рис. 3.7. При увеличении детеныш обнаруживает сходство с целым миром

Во многих местах видно, что исследуемые нами структуры расположены точно в том месте, где сходятся две спирали. Рассмотрим одно такое место (обозначенное как «Рис. 3.6» на рис. 3.5) с дополнительным 30-кратным увеличением. Посмотрите-ка: в самой середине теперь виднеется странный объект, в котором, однако, есть что-то знакомое. Увеличим изображение еще в шесть раз (рис. 3.7) – появляется крохотный дочерний объект, практически идентичный всей структуре! При более внимательном рассмотрении обнаруживаются некоторые отличия присоединенных к этой субструктуре волокон от тех, что выходят из основной структуры, – новые волокна, закручиваясь, уходят на значительно бóльшие относительные расстояния. И при этом маленькое существо выглядит почти неотличимым от своего родителя, – у него даже есть аналогично расположенные собственные детеныши. Можно было бы исследовать и их, если вновь повысить увеличение приборов. «Внуки» тоже будут напоминать своего общего предка – и нетрудно увидеть, что так может продолжаться до бесконечности. Этот странный мир Тор'Блед-Нам можно исследовать как угодно долго, постоянно увеличивая разрешающую способность нашей системы наблюдения. И тогда перед нами предстанет бесконечное разнообразие: никакие две области не являются в точности одинаковыми, но всем им свойственны общие черты, которые очень быстро становятся узнаваемыми. Знакомые нам уже жукообразные существа появляются на всё меньших и меньших масштабах. Каждый раз при этом расположенные рядом волокнистые структуры отличаются от предыдущих, демонстрируя новые фантастические сцены невероятной сложности.

В какой же странной и удивительно замысловатой по своей структуре стране мы оказались? Не сомневаюсь, что многие читатели уже знакомы с ней, но не все. Это не что иное, как фрагмент абстрактной математики – множество, известное под названием множества Мандельброта.[154] При всей его несомненной сложности оно получается на редкость простым образом! Чтобы как следует объяснить правила построения этого множества, необходимо сначала рассказать о том, что такое комплексные числа. Именно этим я сейчас займусь. Комплексные числа нам понадобятся и в дальнейшем. Они являются неотъемлемой частью структуры квантовой механики и вследствие этого лежат в основе поведения самого мира, в котором мы живем. Кроме того, комплексные числа являют собой одно из великих чудес математики. Чтобы объяснить, что такое комплексные числа, мне сначала потребуется напомнить вам, что подразумевается под термином «действительные числа». Не лишним будет также отметить связь этого понятия с действительностью «реального мира»!

§3.2. Действительные числа

Напомним, что натуральные числа являются целыми величинами:

[154] См. Мандельброт [1986]. Выбранная мною конкретная последовательность коэффициентов увеличения взята из работы Пайтгена и Рихтера [1986], в которой можно познакомиться с большим количеством цветных изображений множества Мандельброта. Другие поразительные иллюстрации можно найти в книге Пайтгена и Заупе [1988].

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, ...[155] (*1)[156]

Это самый элементарный и фундаментальный вид чисел. Ими можно количественно измерить любую дискретную сущность[157]: можно говорить о двадцати семи овцах в поле, двух вспышках молнии, двенадцати ночах, тысяче слов, четырех беседах, нуле новых идей, одной ошибке, шести отсутствующих, двукратной смене направления и т.д. Натуральные числа можно складывать или перемножать, получая при этом новые натуральные числа.[158] Мы использовали эти числа при обсуждении алгоритмов в предыдущей главе.

Тем не менее некоторые важные математические операции могут всё же вывести нас за пределы мира натуральных чисел. Простейшая из них – вычитание. Для систематического определения вычитания нам понадобятся отрицательные числа.[159] Теперь мы можем выстроить всю систему целых чисел:

... –6, –5, –4, –3, –2, –1, 0, 1, 2, 3, 4, 5, 6, 7, ... (*2)

Некоторые вещи – такие, как электрический заряд, банковские балансы или даты,[160] измеряются количественно этими числами. Однако сфера применения целых чисел всё же слишком ограничена, поскольку деление одного числа на другое может оказаться неразрешимой задачей в рамках целых чисел.[161] Соответственно, нам понадобятся дроби, или, как их называют, рациональные числа:

$$0,\ 1,\ -1,\ \frac{1}{2},\ -1,\ 2,\ -2,\ \frac{3}{2},\ -\frac{3}{2},\ \frac{1}{3},\ \ldots\ .$$

Этих чисел достаточно для операций конечной арифметики, но для очень многих задач нам потребуется пойти еще дальше, с тем чтобы охватить бесконечные операции или операции перехода к пределу. Например, хорошо известная – и играющая огромную роль в математике – величина π возникает как результат многих бесконечных выражений.[162] В частности, мы имеем:

---

155 **В.Э.:** Разумеется, в этой строке приведены нотаты чисел, а не сами числа.

156 **В.Э.:** Красные метки добавлены мной для последующей ссылки на эти строки.

157 **В.Э.:** То есть – отнести ее саму к тому или иному таксону классификации N_N или ее соотношение с «единичной сущностью» к таксону классификации N_R.

158 **В.Э.:** Для каждой отдельной «арифметической операции», например 2 × 2, вместо пенроузовских слов точнее будет сказать так: 1) первичными действиями мы можем убедиться, что если количество «сущностей» относится к таксону 2, и количество элементов в них тоже к таксону 2, то общее количество элементов будет относиться к таксону 4; 2) вторичными же действиями мы можем сопоставить нотатам «2 × 2» нотату «4». Действия этих двух видов нужно видеть за словами «*натуральные числа можно складывать или перемножать*», а если ЭТО не видеть, то уже и не знаешь, что на самом деле происходит.

159 **В.Э.:** В традиционных изложениях «необходимость» в отрицательных числах появляется из-за того, что «нужно» «систематически определить» вычитание. Пенроуз просто повторяет традиционный подход. Однако этот подход целиком находится в рамках вторичных операций и не затрагивает первичные. Применение такого подхода свидетельствует о том, что связь (математики) с «реальным миром» уже утрачена. А в области первичных операций здесь дело обстоит так, что вдобавок к самим «сущностям» появляется понятие их движения «туда» или «назад» (т.е. линейной ориентации). Например, вавилонский купец закупает зерно (направление +) и продает его (направление –). Пока он продает меньше, чем покупает, «операция вычитания» четко определена. Но когда он продал больше, чем закупил, то получается «отрицательная сущность». Это может означать, например, что он должен покупателю, который ему уже заплатил, но товар еще не получил. В любом случае отрицательные числа в области первичных операций появляются тогда, когда в «сущностях» (т.е. множествах) введена линейная ориентация.

160 На самом деле при счете дат имеет место некоторое отступление от этого правила, поскольку нулевой год пропускается.

161 **В.Э.:** Опять дроби вводятся по традиционному приему исключительно с точки зрения вторичных операций (т.е. при утраченной связи с физическим миром). А чтобы эту связь не утрачивать, не надо выпускать из виду первичные операции. В первичных операциях речь идет о соотношениях множеств.

162 **В.Э.:** Как хорошо известно из школьной математики, число π первоначально вводится как соотношение длины окружности к диаметру. (Даже само обозначение π представляет собой первую букву греческого слова «окружность»). То есть, если мы рассматриваем окружность как некоторое множество точек П, а диаметр как другое множество точек Д, то π будет тот таксон, к которому программа R отнесет соотношение множеств П/Д. С этой точки зрения π ничем не отличается от «рационального» числа. Беда только в том, что П и Д не являются конкретными множествами, и мы не знаем, сколько в них элементов. П и Д являются абстрактными множествами – потенциальными продуктами программы обведения окружности вокруг центра. Так дела с числом π обстоят в области первичных операций. В области же вторичных операций мы можем обнаружить, что некоторые (бесконечные) вычислительные процессы,

$$\pi = 2\left(\frac{2}{1} \times \frac{2}{3} \times \frac{4}{3} \times \frac{4}{5} \times \frac{6}{5} \times \frac{6}{7} \times \frac{8}{7} \times \frac{8}{9} \times \dots\right),$$

а также

$$\pi = 4\left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \frac{1}{11} + \dots\right).$$

Это знаменитые выражения. Первое из них было найдено английским математиком, филологом и криптографом Джоном Уоллисом в 1655 году, а второе – шотландским математиком и астрономом (а также изобретателем первого телескопа-рефлектора) Джеймсом Грегори в 1671 году. Как и π, определенные подобным образом числа <u>не</u> обязаны быть рациональными (то есть представляться в виде m/n, где m и n – целые числа, причем n не равно нулю). Систему чисел необходимо <u>расширить</u>, обеспечив возможность включения в нее таких величин.

Расширенная таким образом система чисел называется системой <u>действительных чисел</u> – тех самых хорошо знакомых нам чисел, что представляются в виде бесконечных десятичных дробей, таких как:

–583,70264439121009538... .

В этом представлении мы получаем следующее известное выражение для числа π:

π = 3,14159265358979323846....

Другими примерами чисел, представимых таким образом, являются квадратные корни (или кубические корни, или корни четвертой степени) из положительных рациональных чисел, такие как:

√2= 1,41421356237309504...

или же квадратные корни (или кубические корни и т.д.) любого положительного числа, как, например, выражение для числа π, найденное великим швейцарским математиком Леонардом Эйлером:

$$\pi = \sqrt{6\left(1 + \frac{1}{4} + \frac{1}{9} + \frac{1}{25} + \frac{1}{36} + \dots\right)}.$$

Действительные числа нам в сущности хорошо знакомы – мы с ними сталкиваемся в повседневной жизни. Правда обычно нас интересуют всего лишь приближения к этим числам и мы предпочитаем ограничиваться разложениями, состоящими из небольшого числа десятичных знаков. Тем не менее, в математических утверждениях может потребоваться <u>точное</u> задание действительных чисел и, как следствие, необходимость в некотором бесконечном способе описания наподобие бесконечной десятичной дроби, или какого-нибудь иного бесконечного математического выражения вроде приведенных выше формул для числа π, предложенных Уоллисом, Грегори и Эйлером. (В дальнейшем я буду обычно использовать десятичные дроби, но лишь потому, что они нам наиболее привычны. У математиков есть множество разных и более удовлетворительных способов представления действительных чисел, но нас это здесь не интересует.)

Может создаться впечатление, что представить себе всё бесконечное десятичное разложение целиком невозможно, но это не так. Вот простой пример, когда вся последовательность знаков оказывается явным образом обозримой:

1/3 = 0,333333333333333... .

Многоточие указывает на то, что последовательность троек продолжается бесконечно. Для получения полного представления об этом разложении достаточно знать, что оно действительно состоит из неограниченной последовательности одних лишь троек. У каждого рационального числа есть повторяющееся (или конечное) десятичное представление вроде:

93/74 = 1,2567567567567567...,

где последовательность 567 повторяется неограниченное число раз. Это число тоже оказывается полностью обозримым. Также обозримым является выражение

работающие с нотатами чисел, (три из них приводит Пенроуз: процессы Уоллиса, Грегори и Эйлера) будут приходить к одному и тому же пределу, который мы имеем основания считать числом π. А основания эти возникают из того, что эти (вторичные) вычислительные процессы изоморфны (первичным) конструкторским процессам (таким, как вписывание многогранников в окружность, всё более и более приближающихся к самой окружности).

0,220002222000000222222000000002222222220...,

которое определяет иррациональное число[163] (оно просто состоит из последовательностей нулей и двоек, длины которых каждый раз увеличиваются на единицу), и еще много похожих выражений. В каждом таком случае нам достаточно знать правило, по которому составлено разложение. Знание алгоритма порождения очередной цифры в разложении числа – при условии, что такой алгоритм существует – дает нам способ «увидеть» целиком всё бесконечное десятичное разложение. Действительные числа с алгоритмически порождаемыми десятичными разложениями называются вычислимыми числами (см. также с. 56). (При этом не важно, десятичное это разложение или двоичное. Вычислимыми в этом смысле оказываются одни и те же числа, независимо от использованного основания разложения.) Только что рассмотренные числа π и $\sqrt{2}$ представляют собой примеры вычислимых чисел. В обоих случаях подробное описание соответствующего правила – задача довольно-таки кропотливая, но, в принципе, нетрудная.

Есть, однако, действительные числа, которые не являются вычислимыми в упомянутом выше смысле.[164] Как мы убедились в главе 2, существуют невычислимые и при этом совершенно четко определенные последовательности. В качестве примера можно рассмотреть десятичное разложение, в котором n-я цифра равна 0 или 1 в зависимости от того, останавливается или нет n-я машина Тьюринга,[165] производящая действия над числом n. В общем случае мы потребуем лишь, чтобы для действительного числа существовало какое-нибудь бесконечное десятичное

---

[163] **В.Э.:** Здесь уже видно, что Пенроуз утратил различие между самим числом и его нотатой. Для него это становится одно и то же. Если в строках (*1) или (*2) пренебрежение этим различием можно было считать продиктованным соображениями простоты и краткости изложения, то здесь так считать уже невозможно. Здесь Пенроуз «наобум» конструирует какую-то нотату и считает ее «числом». Раньше у нас сначала существовало число (таксон классификации множеств), потом оно как-то обозначалось нотатой: «IV», «4» или как-то еще. Теперь же дело обстоит наоборот: сначала есть нотата (обозначение), и из того, что существует обозначение, делается вывод, что существует и то, что оно обозначает. (Хороший логический прием! Попробуйте, читатель, применить его в повседневной жизни, например, к обозначениям «бес», «куздра» или «тамизедун»).

[164] **В.Э.:** Здесь рассказ Пенроуза стал уже настолько расплывчатым, что нам обязательно нужно остановиться и разобраться. Числа (согласно Веданской теории) есть таксоны классификации множеств или их соотношений по программам N, R, R′, R″ (можем оставить одну лишь R′, если говорим только о тех числах, о которых сейчас рассуждает Пенроуз). Чтобы число было определено, нужно определить два множества, И и Е (измеряемое и единицу). Если мы это можем сделать, то «число существует». (Аналогично, как для числа π мы определяли множества точек П – окружность и Д – диаметр). Пенроуз же полностью перешел от множеств (первичной области) к цифрам (вторичной области) и занялся исключительно конструированием нотат. Хотя такой прием с логической точки зрения и является довольно рискованным (вспомните обозначения «бес», «куздра» и «тамизедун» и соответствующие им объекты – соответствующие ли?), но всё же при определенной натяжке мы можем предположить, что при (произвольно) заданной нотате мы можем далее конструировать и такие множества, соотношению которых данная нотата будет соответствовать. В случае «иррациональных» чисел этот способ построения множеств из нотаты остается довольно туманным, но в принципе ситуация всё же знакомая и допустимая: мы предполагаем, что есть некоторая программа P, строящая множества по нотате; мы не знаем ее алгоритма, но путем бокоанализа создали номиналию (а далее, реалию) ее потенциальных продуктов, и соотношение этих множеств тогда и есть искомое число. Поэтому не будем возражать Пенроузу, когда он, сконструировав нотату, утверждает, что создал число. Но далее Пенроуз начинает говорить о «невычислимых последовательностях», т.е. о «невычислимых числах». Теперь не только множества, определяющие число, не известны, но не известна даже и нотата! Однако нотат, которые вообще нельзя было бы построить, не существует. Любая нотата может быть создана при помощи «машины Луллия» L: путем простого комбинирования цифр – правда, не индивидуально, а вкупе со всеми остальными нотатами. (Этот вопрос подробно и исчерпывающе обсуждался в «Канториане» {МОИ № 38, № 39}). В этом смысле никаких «невычислимых чисел» не существует. (А другие смыслы разберем далее по ходу рассказа Пенроуза).

[165] **В.Э.:** Здесь, значит, мы имеем ситуацию, когда нет не только пары множеств И/Е, но нет даже и нотаты. Вместо нотаты имеется некоторая программа (алгоритм) П, которая будет в нотату ставить 0 или 1 в зависимости от того, что ей выдаст вызываемая ею подпрограмма P (алгоритм которой, разумеется, не известен), определяющая, остановятся ли все n ($n \to \infty$) машины Тьюринга T_n (конструкция которых для данного n, разумеется, не известна, так как она зависит от множества факторов: см. пенроузовскую главу 2!). (Замечательная точность рассуждений!). Из-за неопределенности P и T_n мы, разумеется, не знаем, какова именно будет эта искомая нотата, но точно знаем, что в любом случае она будет среди продуктов «машины Луллия» L. Такова истинная ситуация, (весьма неудачно) характеризуемая Пенроузом словами «*невычислимые и при этом совершенно четко определенные последовательности*».

разложение. Мы не только не требуем существования алгоритма порождения n-й цифры, но нам даже не обязательно знать о существовании какого бы то ни было правила, в принципе определяющего n-ю цифру.[166] Заметим, что вычислимые числа неудобны в работе. Невозможно обойтись одними лишь вычислимыми операциями, даже оперируя вычислимыми числами. Например, в общем случае вычислимым образом невозможно даже решить, равны ли два вычислимых числа друг другу! По этой причине мы будем работать со всеми действительными числами, когда десятичная последовательность может быть любой, а не только, скажем, вычислимой.

В заключение отметим также тождественность действительных чисел, чьи десятичные разложения заканчиваются бесконечной последовательностью девяток, и чисел, чьи разложения заканчиваются бесконечной последовательностью нулей. Например:

$$-27{,}1860999999... = -27{,}1861000000... .$$

§3.3. Сколько же всего действительных чисел?

Давайте остановимся на минутку, чтобы оценить всю колоссальность обобщения при переходе от рациональных чисел к действительным.

Вначале может показаться, что целых чисел больше, чем натуральных, поскольку каждое натуральное число является целым, в то время как некоторые целые числа (а именно отрицательные) натуральными не являются. Аналогично может создаться впечатление, что дробей больше, чем целых чисел. Однако это не так.[167] Согласно мощной и очень красивой теории бесконечных чисел, разработанной в конце XIX века Георгом Кантором – исключительно самобытным немецким математиком русского происхождения,[168] – общее число дробных чисел, общее количество всех целых чисел и число всех натуральных чисел равны одному и тому же бесконечному числу, обозначаемому $\aleph_0$ («алеф-нуль»). (Удивительно, что похожая идея была частично предвосхищена еще за 250 лет до этого в начале XVII века великим итальянским

---

166 Насколько мне известно, точка зрения, согласно которой для любого действительно (действительного? – *В.Э.*) числа должно существовать некое – пусть неэффективное и даже совершенно неопределимое в рамках заданной формальной системы (см. главу 4) – правило, позволяющее определить его n-й знак, является вполне непротиворечивой, хотя и нетрадиционной. Я сильно надеюсь на то, что этот подход действительно непротиворечив, поскольку именно этой точки зрения я сам больше всего хотел бы придерживаться!

167 **В.Э.:** Больше или не больше целых чисел, чем натуральных, и больше или не больше дробных чисел, чем тех же натуральных, – это зависит от того, как мы здесь определяем понятия «больше» или «одинаково», как мы измеряем их «количество». Классическая математика, создавшая алмазный фонд этой науки – дифференциальное и интегральное исчисление – всегда рассматривала математические соотношения как результаты некоторых процессов (функций). Если функция $\varphi(x)$ и функция $\psi(x)$ при $x \to \infty$ обе тоже стремились к бесконечности, то соотношение $\varphi(x) / \psi(x)$, т.е. соотношение ∞/∞ отнюдь не предполагалось обязательно равным 1, а определялось по (золотому!) правилу Лопиталя соотношением производных функций. Если мы сохраняем этот дух классической математики и смотрим на числа как на результат некоторых процессов (а они и есть потенциальные продукты программ, т.е. процессов!), то $\infty/\infty \neq 1$, а $\infty/\infty = 2$ для соотношения целых и натуральных чисел и $\infty/\infty = \infty$ для соотношения дробных и натуральных чисел. «Великое открытие» Кантора состояло в том, что он вводом своего «взаимно-однозначного соответствия» установил, что $\infty/\infty = 1$ (всегда!), то есть, он принял такой постулат и тем самым отступился от духа классической математики, пришедшей в этом месте к правилу Лопиталя уже примерно за 200 лет до Кантора. Оценку этому постулату Кантора мы дадим ниже.

168 **В.Э.:** Георг Кантор был сыном датчанина (лютеранина) и немки (католички), родившимся в России, в Санкт-Петербургской колонии иностранных торговцев, но в 11 лет увезенный родителями в Германию, где и прожил всю остальную жизнь. Кантор был психически болен, страдал маниакально-депрессивным психозом; в депрессивной фазе он прекращал работу и ему давали отпуска на его профессорской работе в Галльском университете и помещали в психиатрическую клинику (каковая имелась при том же университете), а в маниакальной фазе своей болезни Кантор сочинял разные неправильные «теории», например теорию бесконечных множеств и теорию о том, что Шекспира не существовало, а его сочинения написал Бэкон (в ранней стадии канторовской болезни) или что не существовало ни Шекспира, ни Бэкона, а их сочинения написал какой-то неизвестный тайный гений (в поздней стадии канторовской болезни). Болезнь Кантора может вызвать сочувствие, но сочувствия не может вызвать тот факт, что маниакальный бред психически больного человека был принят в качестве «научной истины» такой наукой как математика (!!!). Георг Кантор умер в психиатрической клинике своего университета.

физиком и астрономом Галилео Галилеем. Мы вспомним о некоторых других достижениях Галилея в главе {5}.) Равенство количества целых чисел количеству натуральных чисел видно из следующего взаимно-однозначного соответствия:

Целые числа		Натуральные числа
0	$\leftrightarrow$	0
-1	$\leftrightarrow$	1
1	$\leftrightarrow$	2
-2	$\leftrightarrow$	3
2	$\leftrightarrow$	4
-3	$\leftrightarrow$	5
3	$\leftrightarrow$	6
-4	$\leftrightarrow$	7
.	.	.
.	.	.
.	.	.
$-n$	$\leftrightarrow$	$2n-1$
n	$\leftrightarrow$	$2n$
.	.	.
.	.	.
.	.	.

Обратите внимание,[169] что каждое целое число (в левом столбце) и каждое натуральное число (в правом столбце) встречаются один и только один раз в своем списке. В канторовской теории множеств именно существование такого рода взаимно-однозначного соответствия устанавливает факт равенства числа объектов в левом столбце числу объектов в правом столбце. Таким образом, число целых чисел действительно равно числу натуральных чисел. В данном случае это число бесконечно, но это не имеет значения. (Единственное необычное свойство бесконечных чисел состоит в том, что даже если мы исключим некоторые элементы одного из списков, мы можем установить взаимно-однозначное соответствие между элементами двух списков.) Аналогичным, хотя и несколько более сложным образом, устанавливается взаимно-однозначное соответствие между дробными и целыми числами. (Для этого можно использовать какой-либо из способов представления пар[170] натуральных чисел – числителей и знаменателей – через отдельные натуральные числа; см. главу 2, с.50 {с.62 здесь}). Множества, которые можно поставить во взаимно-однозначное соответствие с рядом натуральных чисел, называются счетными; таким образом, счетные бесконечные множества – это множества, состоящие из $\aleph_0$ элементов. И, как мы только что убедились, множество целых чисел, равно как и множество дробных чисел, является счетным.[171]

Существуют ли множества, не являющиеся счетными? Несмотря на расширение натуральной системы чисел сначала целыми, а затем и рациональными числами, общее число рассматриваемых объектов не увеличилось. Как мы убедились, число объектов во всех случаях осталось счетным. У читателя теперь может создаться впечатление, что все бесконечные множества

[169] **В.Э.:** Больше всего, читатель, обратите внимание на то, что элементы, перечисленные в левом списке, – конечны (т.е. в каждой строке списка имеется только один элемент). Лишь поэтому (а вовсе не потому, что строк в списке недостаточно много!) здесь нельзя проводить знаменитый «диагональный процесс» и обнаруживать ценнейший вывод, что не все целые числа перенумерованы натуральными числами.

[170] **В.Э.:** Обратите внимание, читатель, что теперь в перенумерованном списке в каждой его строке содержится уже пара элементов, т.е. их два. Это больше, чем в предыдущем случае, но всё же еще всего лишь конечное число, и великолепный диагональный процесс пока провести невозможно.

[171] **В.Э.:** На самом деле «счетны» ВСЕ бесконечные множества. Кантор установил постулат, что $\infty/\infty \equiv 1$, и несчетных множеств не существует. Иллюзию существования «несчетных» множеств создает только «диагональный метод». А этот процесс можно будет проводить, как только количество элементов в строке «пронумерованного списка» станет бесконечным. Обратите внимание читатель: возможность проведения диагонального процесса не имеет никакого отношения к количеству строк в списке (о котором после этого процесса будут делаться выводы), а имеет отношение только к количеству элементов в самой строке.

счетны. Это не так, поскольку ситуация меняется коренным образом[172] при переходе к действительным числам. Одним из замечательных достижений Кантора явилось доказательство того, что действительных чисел больше, чем натуральных. При этом Кантор применил так называемый <u>диагональный процесс</u>, который упоминался в главе 2 и который Тьюринг использовал в своем доказательстве неразрешимости проблемы остановки для машин Тьюринга. Доказательство Кантора, как и более позднее доказательство Тьюринга, – это <u>доказательство от противного</u>. Предположим, что утверждение, справедливость которого мы хотим установить, на самом деле ложно, то есть множество действительных чисел счетно. Тогда множество действительных чисел в интервале от 0 до 1 должно быть заведомо счетным и должен существовать <u>какой-нибудь</u> список,[173] устанавливающий взаимно-однозначное соответствие между рассматриваемым множеством действительных чисел и множеством натуральных чисел, наподобие вот этого:

Натуральные числа		Действительные числа
0	↔	0,**1**0357627183...
1	↔	0,1**4**329806115...
2	↔	0,02**1**66095213...
3	↔	0,430**0**5357779...
4	↔	0,9255**0**489101...
5	↔	0,59210**3**43297...
6	↔	0,636679**1**0457...
7	↔	0,8705007**4**193...
8	↔	0,04311737**8**04...
9	↔	0,786350811**5**0...
10	↔	0,4091673889**1**...
.	.	.
.	.	.
.	.	.

Жирным шрифтом выделены диагональные десятичные знаки. В данном случае эти цифры равны:

1, 4, 1, 0, 0, 3, 1, 4, 8, 5, 1, ...

Метод диагонального процесса состоит в построении действительного числа (в интервале от 0 до 1), чье десятичное разложение (после десятичной запятой) отличается в каждом разряде от соответствующего числа приведенной выше последовательности. Для определенности положим, что цифра данного разряда равна 1, если цифра соответствующего разряда на диагонали отлична от 1, и равна 2, если цифра на диагонали равна 1. Таким образом, в рассматриваемом случае получается такое действительное число:

0,21211121112... .

Это действительное число не может быть в списке,[174] поскольку оно отличается от первого числа в первом десятичном разряде (после десятичной запятой), от второго числа – во втором разряде,

---

[172] **В.Э.:** Верно! Ситуация меняется коренным образом: теперь в каждой строке стало бесконечное количество элементов (цифр)! Ур-ра-а-а!!! Теперь можно проводить диагональный процесс! – Но что менялось-то? Количество строк? Или длина строки? Ясно, что решающим было изменение длины строки. А выводы будут делаться о количестве строк! Но, читатель, это ведь логическая ошибка чрезвычайной грубости: выводы, которые должны относиться к <u>длине</u> строки, переносить на <u>количество</u> строк!

[173] **В.Э.:** Однако элементарно очевидно, что этот список, если он претендует на то, что он содержит ВСЕ «действительные числа» (точнее, их нотаты) указанного интервала, то он должен иметь вправо длину n цифр, а вниз длину 10^n строк. Если он имеет не такое соотношение длин, то (вариант 1) он <u>не содержит</u> все требуемые «числа», а если он содержит все «числа», то (вариант 2) соотношение длин <u>таково</u> ($n/10^n$). Теперь выбирайте, мистер Пенроуз, какой вариант Вы подразумеваете? (Или никакой не подразумеваете: в тумане пребываете?). Постулат Кантора ($\infty/\infty \equiv 1$) заставляет Вас полагать, что $n/10^n = 1$ и что $10^n = n$, но это неминуемо приведет Вас к противоречиям.

[174] **В.Э.:** Может или не может это «число» быть в списке – это зависит от того, какую концепцию своей таблицы Пенроуз принял из двух вариантов, предложенных ему в предыдущей моей сноске. Если он

от третьего числа – в третьем разряде и т.д. Таким образом, мы приходим к противоречию,[175] поскольку полагали, что рассматриваемый список содержит все действительные числа в интервале от 0 до 1. Из этого противоречия следует истинность утверждения, которое нам требовалось доказать, – а именно, что не существует взаимно-однозначного соответствия между множеством действительных чисел и множеством натуральных чисел и, соответственно, что число действительных чисел больше[176] числа рациональных чисел и не является счетным.

Число действительных чисел равно бесконечному числу, обозначаемому C. (Здесь C является сокращенным обозначением слова континуум – другого названия системы действительных чисел.) Может возникнуть вопрос, почему мы не обозначаем это число, например, $\aleph_1$. Символ $\aleph_1$ на самом деле обозначает следующее за $\aleph_0$ бесконечное число, а вопрос о том, верно ли утверждение $C = \aleph_1$ – это так называемая континуум-гипотеза, – представляет собой знаменитую и пока что нерешенную проблему.[177]

При этом следует отметить, что множество вычислимых чисел счетно.[178] Пересчитать их можно просто перечислив по порядку машины Тьюринга, порождающие действительные числа (то есть машины, последовательно порождающие цифры каждого разряда действительных чисел). При этом можно исключить из списка любую машину Тьюринга, порождающую действительное число, которое уже встречалось ранее в списке. Поскольку множество машин Тьюринга счетно, то, следовательно, счетным также должно быть и множество вычислимых действительных чисел. Почему же нельзя применить диагональный процесс к этому списку с тем, чтобы породить новое не включенное в список вычислимое число? Ответ состоит в том, что в общем

принял концепцию (1) (матрица квадратна), то полученного «числа» действительно нет в таблице, и этот результат опровергает предположение, что в квадратной матрице могут содержаться ВСЕ указанные числа. (К соотношению количества натуральных и действительных чисел этот результат никакого отношения не имеет). Если же Пенроуз принял концепцию (2) (длина матрицы вправо n цифр, а вниз – 10^n строк), то диагональный процесс охватил не всю таблицу, а только ее верхнюю часть, и «построенный» диагональным процессом элемент имеется в таблице, – но только в неохваченной диагональным процессом ее части. Так что надо просто мыслить четко и ясно, выбирать какой-нибудь один определенный вариант, а не пребывать в «суперпозиции» обоих вариантов сразу.

[175] **В.Э.:** Самое главное, фундаментальное, противоречие кроется, конечно, в собственно постулате Кантора о том, что $\infty/\infty \equiv 1$. Нет никакой необходимости вводить этот постулат. Ситуация предельно просто объясняется и ясна «как божий день» без него – и по «лезвию Оккама» этот постулат должен лететь в мусорную яму. Но вместо этого на его основе строится запутанное, уродливое здание некоего подобия «теории», не соответствующей никакой реальности ни физического, ни платоновского мира, «теории», принципиально лишенной возможности когда-либо быть где-нибудь примененной и приносить какую-либо пользу людям (за исключением, разумеется, тех профессоров, которые ею занимаются, выдавая это за «научную деятельность» и выманивая таким способом деньги от общества).

[176] **В.Э.:** С тем, что действительных чисел больше, чем рациональных, мы можем согласиться по тем же соображениям, по которым мы соглашаемся, что целых чисел больше, чем натуральных (т.е. – ДО введения постулата Кантора о том, что $\infty/\infty \equiv 1$). Однако, если мы рассуждаем в рамках системы, принявшей этот постулат, то у нас НЕТ оснований считать, что действительных чисел больше, чем рациональных. Мнение о таком превосходстве количества опирается, во-первых, на диагональный процесс, но возможность проведения этого процесса зависит НЕ от количества строк в «пронумерованном списке», а от бесконечности длины самих строк, и перенос выводов о длине строк на их количество представляет собой элементарную логическую ошибку. Во-вторых, с бесконечной длиной строк связана и другая особенность «списка действительных чисел»: его нельзя построить по линейному алгоритму, т.е. строя одну строку за другой (как строились списки целых и рациональных «чисел», состоящие из конечных строк). Чтобы таким же образом (одну строку за другой) строить список «действительных чисел», алгоритм должен был бы «перепрыгивать через бесконечность» каждой очередной строки, чтобы приступить к созданию следующей. Это обстоятельство облегчает адептам Кантора пребывать во мнении, что действительных чисел якобы больше рациональных даже если принят постулат Кантора ($\infty/\infty \equiv 1$). Но существуют нелинейные алгоритмы создания «всех действительных чисел», и их изучение показывает, что нет абсолютно никаких оснований полагать, что это множество чем-то принципиально отличается от других бесконечных множеств. **Если принят постулат Кантора, то ВСЕ бесконечности одинаковы**. (И, следовательно, нет ни необходимости, ни смысла этот постулат принимать).

[177] **В.Э.:** Если мы о канторовском диагональном процессе знаем то, что мы теперь знаем, то «проблема континуума» сводится к вопросу: «Есть ли какое-нибудь промежуточное значение между *конечно* и *бесконечно* в отношении длины строк той матрицы, в которой мы (вслед за Пенроузом) проводили диагональный процесс?». (Великая научная проблема – ничего не скажешь!)

[178] **В.Э.:** Поскольку на самом деле множество действительных чисел тоже «счетно», то данное утверждение Пенроуза не представляет интереса.

случае невозможно с помощью вычислений решить, следует ли ту или иную машину Тьюринга включать в список, поскольку для этого мы должны были бы иметь возможность решить проблему остановки. Некоторые машины Тьюринга, начав порождение цифр действительного числа, могут зависнуть и оказаться уже не в состоянии выдать очередную цифру (поскольку они «не остановятся»). Не существует вычислимого способа, который позволил бы решить, какие именно машины Тьюринга зависнут таким образом. Это, в сущности, и есть проблема остановки. Значит, хотя метод диагонального процесса и породит некоторое действительное число, последнее не будет вычислимым. На самом деле, это рассуждение может использоваться для доказательства существования невычислимых чисел. Именно в этом ключе выдержано описанное в предыдущей главе тьюринговское доказательство существования классов алгоритмически неразрешимых задач. Другие области применения диагонального процесса будут рассмотрены дальше.

§3.4. «Действительность» действительных чисел

Если отвлечься от понятия вычислимости, то действительные числа называются «действительными», потому что они, как представляется, дают величины, необходимые для измерения[179] расстояний, углов, времени, энергии, температуры и многих других геометрических и физических параметров. Однако связь абстрактно определенных «действительных» чисел с физическими величинами не так проста, как может показаться. Действительные числа следует рассматривать скорее как некоторую математическую идеализацию, чем как реальную меру физически объективных величин. Система действительных чисел обладает, например, таким свойством, что между любыми двумя действительными числами (вне зависимости от их близости) существует третье действительное число. При этом совершенно не ясно, можно ли обоснованно утверждать то же самое о физических расстояниях или промежутках времени. Если мы продолжим дробить физическое расстояние между двумя точками, то мы в конце концов достигнем масштабов столь малых, что само понятие расстояния в обычном его смысле станет бессмысленным. Предполагается, что это действительно имеет место на масштабах, характерных для квантовой теории гравитации, которые в 10^{20} раз[180] меньше размеров субатомных частиц. Но чтобы отобразить действительные числа, нам потребуется дойти до сколь угодно более мелких масштабов, которые, например, в 10^{200}, 10^{2000} или даже в $10^{10^{200}}$ раз меньше размеров частиц.[181] И совершенно не ясно, есть ли какой бы то ни было физический смысл у столь абсурдно малых масштабов. То же самое можно сказать и в отношении столь же малых интервалов времени.

Система действительных чисел выбрана в физике в силу ее математической полезности, простоты и изящества, а также поскольку она согласуется на очень широком интервале масштабов с физическими понятиями пространства и времени. Она выбрана не потому, что мы будто бы знаем, что она согласуется с упомянутыми физическими величинами на всех масштабах. Такое согласие вполне может не иметь места на очень малых пространственных и временных масштабах. Обычные расстояния измеряются при помощи линейки, но линейка оказывается «зернистой» при переходе к масштабам образующих ее атомов. Само по себе это не мешает нам продолжать использовать действительные числа подходящим образом, но измерение меньших расстояний требует уже гораздо большей изобретательности. По крайней мере, мы должны быть готовы предположить, что на очень-очень малых масштабах могут встречаться принципиальные трудности с расстояниями. Как оказывается, природа оказалась к нам на удивление благосклонна, сделав те самые действительные числа, которые мы привыкли повседневно применять для описания предметов на макро-масштабах, пригодными для описания расстояний гораздо меньших атомных – по крайней мере, на масштабах, равных одной сотой

---

[179] **В.Э.:** Нет, не поэтому – и странно, что Пенроуз этого не знает! Действительные (Real – реальные) числа исторически получили такое название в противоположность «имагинарным» («воображаемым») числам, таким, как i, –i, 2i и т.д. (А объединения «реальных» и «воображаемых» чисел стали называться «комплексными»).

[180] Напомним, что 10^{20} означает число 100˙000˙000˙000˙000˙000˙000, то есть единицу с двадцатью нулями.

[181] **В.Э.:** Вот, поэтому-то и не годится никакая другая концепция числа, кроме той, которую дает Веданская теория: число – это потенциальный продукт программы, а программу потенциально можно продолжать сколь угодно далеко.

«классического» диаметра элементарной частицы – такой, как электрон или протон, – и, по-видимому, вплоть до «масштабов квантовой теории гравитации», что на двадцать порядков меньше размеров таких частиц! Это пример исключительно сильной экстраполяции нашего опыта. Сфера применимости привычного понятия расстояния, измеряемого действительными числами, по-видимому, простирается до самых далеких квазаров и еще дальше. Общий диапазон измеримых расстояний составляет 10^{42}, а может быть, 10^{60} или даже больше. Кстати, сомнения в правомерности использования системы действительных чисел высказывались не так уж часто. Почему же мы так уверены в том, что эти числа дают точное описание физических явлений, хотя реально об их применимости мы знаем лишь в весьма ограниченном диапазоне масштабов? Должно быть, эта уверенность – возможно, неверная – основывается на (правда, не очень часто признаваемых) логическом изяществе, внутренней согласованности и математической мощи системы действительных чисел в сочетании с верой в глубинную математическую гармонию природы.[182]

§3.5. Комплексные числа

Оказывается, что действительные числа – это не единственная математически мощная и изящная система чисел. Система действительных чисел всё же не лишена некоторых неудобств. Например, квадратные корни можно извлекать только из положительных чисел (или нуля), но никак не из отрицательных чисел.[183] С математической точки зрения – и отвлекаясь пока что от вопроса о непосредственной связи с физическим миром – было бы очень удобно иметь возможность извлекать квадратные корни как из положительных, так и из отрицательных чисел. Давайте постулируем существование, или попросту «изобретем» квадратный корень из числа –1. Обозначим его буквой i. Тогда мы имеем:

$$i^2 = -1.$$

Величина i, конечно же, не может быть действительным числом, поскольку произведение действительного числа на самого себя всегда положительно (или равно нулю, если само число равно нулю). Поэтому числа, квадраты которых отрицательны, обычно называют <u>мнимыми</u>. Следует, однако, отметить, что эти «мнимые» числа не менее реальны, чем ставшие уже привычными «действительные» числа. Как я уже отмечал выше, связь таких «действительных» чисел с <u>физической</u> реальностью далеко не столь непосредственна и убедительна, как может показаться на первый взгляд, и основана на математической идеализации о допустимости бесконечного уточнения, которая не имеет ясного <u>априорного</u> обоснования в природе.

Имея квадратный корень из –1, можно без особого труда получить квадратные корни для всех действительных чисел. Если *a* является положительным действительным числом, то величина

$$i \times \sqrt{a}$$

есть квадратный корень из отрицательного действительного числа –*a*. (У этого числа есть еще другой квадратный корень, а именно $-i \times \sqrt{a}$.) Ну, а что же можно сказать о самом числе i? Есть ли у него квадратный корень? Разумеется есть, поскольку, как легко проверить, величина

$$\frac{1+i}{\sqrt{2}}$$

---

[182] **В.Э.:** Ну, весь этот абзац показывает, что у Пенроуза нет глубокого понимания природы чисел. Концепция Веданской теории по этому вопросу была изложена в Предисловии к настоящему тому.

[183] **В.Э.:** То, что (в системе действительных чисел) невозможно извлечь квадратный корень из отрицательного числа, является следствием того обстоятельства, что $-a \times -a = +a$. В свою очередь, это обстоятельство (вторичных операций) в области первичных операций вытекает из того, что (при линейной ориентации) дважды поменяв направление, мы получаем исходную ориентацию. Если ЭТО понимать, то уже в общем-то нетрудно догадаться, что для того, чтобы при двойном изменении ориентации получить противоположное направление, каждое из этих двойных изменений должно быть не на «полный оборот» (на 180°), а на «пол-оборота» (на 90°). То есть, от линейной ориентации (на прямой) нужно перейти к планарной ориентации (на плоскости). Вот, к этому выводу математика и шла – только долгими, трудными и окольными путями – и дальнейший рассказ Пенроуза показывает, что даже и теперь даже и он не очень-то хорошо понимает глубинную сущность «комплексных чисел». А она проста: никаких таинственных неизвлекаемых корней, никаких «мнимых» чисел; просто ротация – элементарное вращение.

(равно как и та же величина, взятая с отрицательным знаком), будучи возведена в квадрат, равна i. А у этой величины, в свою очередь, есть квадратный корень? Ответ опять положительный: квадрат числа

$$\sqrt{\frac{1+1/\sqrt{2}}{2}}+i\sqrt{\frac{1-1/\sqrt{2}}{2}}$$

или того же числа, взятого с отрицательным знаком, действительно равен $(1 + i)/\sqrt{2}$.

Обратите внимание, что при образовании такого рода величин мы позволили себе складывать действительные и мнимые числа, а также умножать наши числа на произвольные действительные числа (или делить их на произвольные ненулевые действительные числа, а это то же самое, что умножать их на обратные величины). Получаемые таким образом объекты называются комплексными числами. Комплексное число это число вида:

$$a + ib,$$

где a и b – это действительные числа, называемые, соответственно, действительной и мнимой частью комплексного числа. Правила сложения и умножения двух таких чисел вытекают из обычных правил (школьной) алгебры с одним дополнительным правилом $i^2 = -1$:

$$(a + ib) + (c + id) = (a + c) + i(b + d),$$
$$(a + ib) \times (c + id) = (ac - bd) + i(ad + bc).$$

Удивительное дело: к созданию этой системы чисел нас подтолкнуло желание иметь возможность извлечения квадратных корней из любых чисел. Эта цель достигнута, хотя само по себе это еще не очевидно. Но новая система чисел позволяет делать гораздо больше: безнаказанно извлекать кубические корни, корни пятой степени, корни девяносто девятой степени, корни π-й степени, корни степени 1 + i и т.д.[184] (это смог доказать еще в XVIII веке великий математик Леонард Эйлер). В качестве другого примера волшебных свойств комплексных чисел рассмотрим довольно сложные на вид тригонометрические формулы, которые проходят в школе. Так, синус и косинус суммы двух углов

$$\sin(A + B) = \sin A \cos B + \cos A \sin B,$$
$$\cos(A + B) = \cos A \cos B - \sin A \sin B$$

представляют собой, соответственно, просто-напросто мнимую и действительную части гораздо более простого (и легче запоминаемого!) комплексного уравнения[185]:

$$e^{iA+iB} = e^{iA}e^{iB}.$$

Всё, что нам нужно здесь знать, это «формула Эйлера» (по-видимому, полученная за много лет до Эйлера замечательным английским математиком XVI века Роджером Котсом):

$$e^{iA} = \cos A + i \sin A,$$

которую мы теперь подставим в приведенное выше уравнение. В результате имеем:

$$\cos(A + B) + i \sin(A + B) = (\cos A + i \sin A)(\cos B + i \sin B),$$

и, выполнив умножение в правой части, получим искомые тригонометрические соотношения.

Более того, любое алгебраическое уравнение

$$a_0 + a_1z + a_2z^2 + a_3z^3 + ... + a_nz^n = 0$$

---

[184] **В.Э.:** В области первичных операций (над множествами) это означает, что ротация (вращение) является гораздо более совершенным и завершенным видом движения, нежели смена направления «туда» или «сюда». Раз (введением отрицательных чисел) вообще ввели учет ориентации над метрическими числами, то невозможно было уже остановиться, пока не дошли до ротации и не получили наконец совершенную систему чисел.

[185] Величина e = 2,7182818285... (основание натуральных логарифмов, иррациональное число, по своему значению для математики сравнимое с числом π) определяется как

$$e = 1 + \frac{1}{1} + \frac{1}{1\times 2} + \frac{1}{1\times 2\times 3} + \dots,$$

а e^z означает степень z от e, которая равна

$$e^z = 1 + \frac{z}{1} + \frac{z^2}{1\times 2} + \frac{z^3}{1\times 2\times 3} + \dots .$$

(где a_0, a_1, a_2, ..., a_n являются комплексными числами и $a_n \neq 0$) всегда имеет своим решением некоторое комплексное число z. Например, существует комплексное число, удовлетворяющее соотношению:

$$z^{102} + 99z^{33} - \pi z = -417 + i,$$

хотя это совершенно не очевидно!

Это общее свойство иногда называют «основной теоремой алгебры». Многие математики XVIII века старались доказать этот результат. Получить удовлетворительное доказательство в общем случае оказалось не под силу даже Эйлеру. И только в 1831 году великий математик и естествоиспытатель Карл Фридрих Гаусс предложил потрясающий по своей оригинальности ход рассуждений и представил первое общее доказательство. Ключевым компонентом этого доказательства было применение топологических[186] рассуждений к геометрическому представлению комплексных чисел.

На самом деле Гаусс не был первым, кто использовал геометрическое представление комплексных чисел. Уоллис сделал то же самое примерно за двести лет до Гаусса, хотя далеко не столь результативно. Геометрическое представление комплексных чисел обычно связывают с именем Жана Робера Аргана – швейцарского бухгалтера, описавшего это представление в 1806 году, хотя полное описание этого представления было на самом деле дано девятью годами раньше норвежским геодезистом Каспаром Весселем. Согласно этой традиционной (хотя и не совсем правильной с исторической точки зрения) терминологии, я буду называть стандартное геометрическое представление комплексных чисел плоскостью Аргана.

Плоскость Аргана представляет собой обычную евклидову плоскость со стандартными декартовыми координатами x и y, где x обозначает расстояние по горизонтали (положительное вправо и отрицательное влево), а y – расстояние по вертикали (положительное вверху и отрицательное внизу). В этом случае комплексное число

$$z = x + iy$$

представляется точкой на плоскости Аргана с координатами

$$(x, y)$$

(рис. 3.8). Обратите внимание, что число 0 (рассматриваемое как комплексное число) соответствует началу координат, а число 1 – одной из точек на оси x.

Рис. 3.8. Изображение комплексного числа $z = x + iy$ на плоскости Аргана

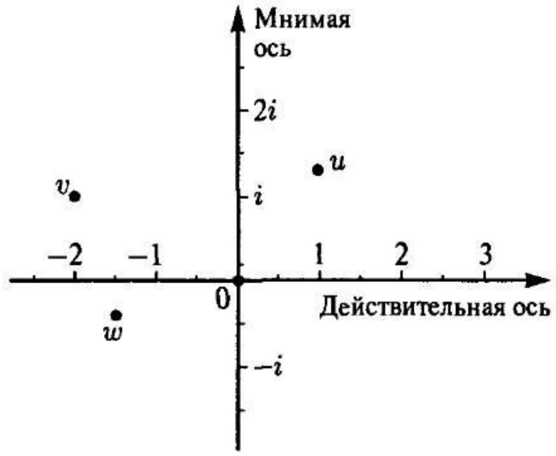


Рис. 3.9. Расположение чисел $u = 1 + i\,1{,}3$, $v = -2 + i$, $w = -1{,}5 - i\,0{,}4$ на плоскости Аргана

Плоскость Аргана есть просто способ геометрически наглядной организации семейства комплексных чисел. Такое представление не является для нас чем-то совершенно новым. Мы уже знакомы с геометрическим представлением действительных чисел – в виде прямой линии, простирающейся на неограниченное расстояние в обоих направлениях. Одна из точек обозначена как 0, а еще одна – как 1. Точка 2 смещена относительно точки 1 ровно настолько, насколько точка 1 смещена относительно точки 0; точка 1/2 расположена в точности посередине между

[186] Слово «топологический» означает, что речь идет о разделе геометрии, – иногда называемом «геометрией резиновой поверхности», – в котором расстояния не имеют никакого значения, а важны только свойства непрерывности объектов.

точками 0 и 1; точка –1 расположена так, что точка 0 находится в точности посередине между точками –1 и 1, и т.д., и т.п. Отображенное таким образом множество действительных чисел называется действительной прямой. В случае комплексных чисел у нас есть уже целых два действительных числа – a и b – которые могут рассматриваться как координаты комплексного числа a + ib. Эти два числа дают нам две координаты точки на плоскости, в данном случае – на плоскости Аргана. Для примера я указал на рис. 3.9 приблизительные положения комплексных чисел

$$u = 1 + \mathrm{i}\,1{,}3,\quad v = -2 + \mathrm{i},\quad w = -1{,}5 - \mathrm{i}\,0{,}4.$$

Теперь основные алгебраические операции сложения и умножения комплексных чисел приобретают ясную геометрическую интерпретацию. Рассмотрим сначала сложение. Предположим, что u и v это два комплексных числа, представленные на плоскости Аргана в соответствии с описанной выше схемой. Тогда сумма этих двух чисел $u + v$ представляется «векторной суммой» двух точек, то есть точка $u + v$ находится на месте недостающей вершины параллелограмма, образованного точками u, v и началом координат 0. Нетрудно убедиться, что эта конструкция (рис. 3.10) действительно дает сумму двух чисел, но соответствующее доказательство я здесь опускаю.

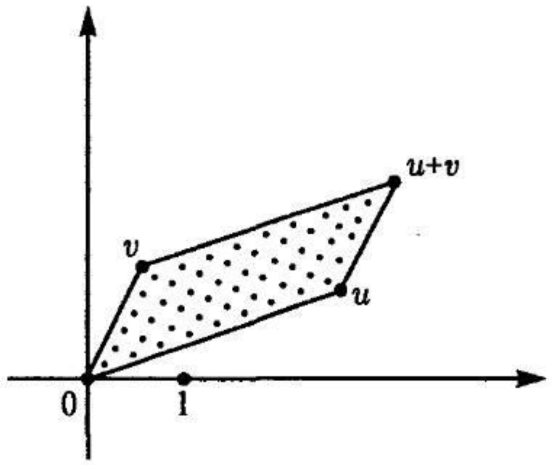


Рис. 3.10. Сумма $u + v$ двух комплексных чисел определяется по правилу параллелограмма

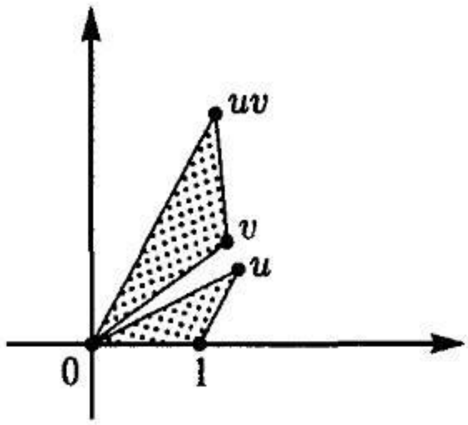


Рис. 3.11. Произведение uv двух комплексных чисел u и v – это такое число, что треугольник, образованный точками 0, v и uv, подобен треугольнику, образованному точками 0, 1 и u. То же самое можно сформулировать иначе: расстояние точки uv от 0 равно произведению расстояний от 0 до точек u и v, а угол между uv и действительной (горизонтальной) осью равен сумме углов между этой осью и отрезками к точкам u и v

Произведение uv двух комплексных чисел тоже имеет простую, хотя и, быть может, несколько менее очевидную геометрическую интерпретацию (рис. 3.11). (Я опять опускаю доказательство.) Угол при начале координат между 1 и uv равен сумме углов между 1 и v и между 1 и u (все углы измеряются против часовой стрелки), а расстояние точки uv от начала координат равно произведению расстояний от начала координат до u и v. Это эквивалентно утверждению, что треугольник, образованный точками 0, v и uv подобен (и ориентирован подобно) треугольнику, образованному точками 0, 1 и u. (Энергичные читатели, не знакомые с такого рода построениями, могут сами убедиться в том, что эти построения непосредственно следуют из только что приведенных алгебраических правил сложения и умножения комплексных чисел, также как и упомянутые выше тригонометрические тождества.)

§3.6. Построение множества Мандельброта

Теперь мы можем рассмотреть, как определяется множество Мандельброта. Пусть z – это некоторое произвольное комплексное число. Каковым бы ни было это число, оно представляется некоторой точкой на плоскости Аргана. Рассмотрим теперь отображение, при котором z превращается в новое комплексное число, равное

$$z \to z^2 + c,$$

где c есть некое фиксированное (то есть заданное) комплексное число. Числу $z^2 + c$ будет сопоставляться некоторая другая точка на плоскости Аргана. Например, если c равно числу 1,63 – i4,2, то z отображается согласно формуле

$$z \to z^2 + 1{,}63 - i4{,}2,$$

так что, в частности, число 3 превратится в

$$3^2 + 1{,}63 - i4{,}2 = 9 + 1{,}63 - i4{,}2 = 10{,}63 - i4{,}2,$$

а число –2,7 + i0,3 в

$$(-2{,}7 + i0{,}3)^2 + 1{,}63 - i4{,}2 = (-2{,}7)^2 - (0{,}3)^2 + 1{,}63 + i\{(-2{,}7)(0{,}3) - 4{,}2\} = 8{,}83 - i5{,}82.$$

Когда числа становятся громоздкими, вычисления лучше выполнять на компьютере.

Теперь, каково бы ни было число c, число 0 превращается, согласно принятой схеме, в число c. А что же можно сказать о самом числе c? Оно превращается в $c^2 + c$. Давайте продолжим этот процесс, применив наше преобразование к $c^2 + c$. Мы получим:

$$(c^2 + c)^2 + c = c^4 + 2c^3 + c^2 + c.$$

Снова повторим отображение, применив его к приведенному выше числу. Мы получим:

$$(c^4 + 2c^3 + c^2 + c)^2 + c = c^8 + 4c^7 + 6c^6 + 6c^5 + 5c^4 + 2c^3 + c^2 + c.$$

Потом еще раз применим процедуру, теперь уже к последнему числу, и т.д. В результате мы получаем последовательность комплексных чисел, которая начинается с числа 0:

$$0,\ c,\ c^2 + c,\ c^4 + 2c^3 + c^2 + c,\ \ldots .$$

Данная процедура, будучи реализована при некоторых определенных значениях комплексного числа c, дает последовательность чисел, которые всё время остаются вблизи начала координат плоскости Аргана; точнее, для выбранных таким образом значений c получаемая последовательность оказывается ограниченной, то есть любой ее член находится в пределах некоторого фиксированного круга с центром в начале координат (рис. 3.12). Хорошим примером здесь может служить последовательность $c = 0$, поскольку каждый ее член равен 0. Другим примером ограниченного поведения является случай $c = 1$,[187] при котором получается последовательность 0, –1, 0, –1, 0, –1, ...; еще один пример – это c = i, когда получается последовательность 0, i, i–1, –i, i–1, –i, i–1, –i, ... Однако, для целого ряда других комплексных чисел c получаемая последовательность всё дальше удаляется от начала координат, то есть является неограниченной и не может находиться целиком в пределах фиксированного круга. Именно так происходит при $c = 1$, когда получается последовательность 0, 1, 2, 5, 26, 677, 458˙330, ... ; аналогичное поведение имеет место в случае $c = 3$ – соответствующая последовательность имеет вид 0, –3, 6, 33, 1086, ...; а также случай c = i–1, который приводит к последовательности 0, i–1, –i–1, –1 + 3i, –9 – i5, 55 + i91, –5257 + i10011, ...

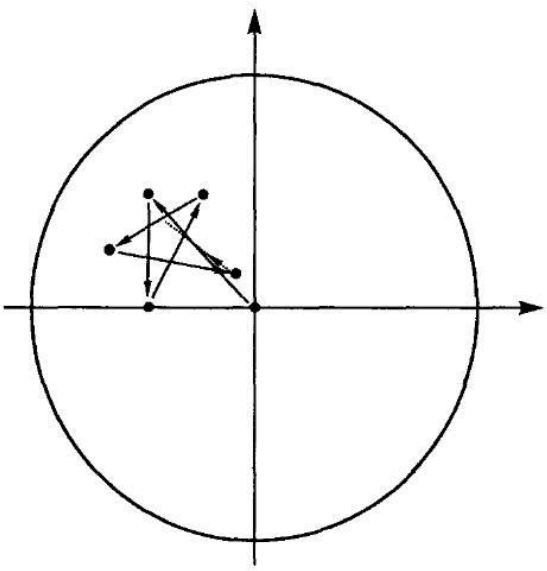


Рис. 3.12. Последовательность точек на плоскости Аргана ограничена, если вся она целиком помещается в пределах некоторого фиксированного круга. (Итерация на рисунке начинается с точки 0 и построена для $c = -1/2 + (1/2)$ i.)

Множество Мандельброта – то есть зачерненная часть страны Тор'Блед-Нам[188] – как раз и есть та самая область на плоскости Аргана, что состоит из всех точек c, для которых получаемая последовательность является ограниченной. Белая же область состоит из тех точек c, для которых получается неограниченная последовательность. Приведенные выше подробные рисунки основаны на результатах компьютерных вычислений. На компьютере был проведен систематический перебор всевозможных комплексных чисел c, для каждого из них строилась

[187] **В.Э.:** Так в книге. Видимо, должно быть: $c = -1$.

[188] В оригинале – Tor'Bled-Nam. А что получится, если прочитать наоборот? – *Прим. ред.*

последовательность 0, c, $c^2 + c$, ..., после чего согласно некоторому критерию определялось, ограничена или нет получаемая последовательность. Если последовательность оказывалась ограниченной, то соответствующая числу c точка экрана становилась черной. Таким образом, для каждой точки в рассматриваемой области компьютер решал, закрасить ее в белый или черный цвет.

Множество Мандельброта впечатляет своей сложностью, особенно учитывая, как это часто бывает в математике, удивительную простоту его определения. Кроме того, структура этого множества в целом не очень чувствительна к выбору алгебраической формы отображения – $z \to z^2 + c$. Многие другие итеративные отображения (например, $z \to z^3 + iz^2 + c$) приводят к поразительно похожим структурам (при условии выбора подходящего начального числа – возможно, это не 0, а значение, четко задаваемое вполне определенным математическим правилом для каждого разумно выбранного отображения). Подобные «мандельбротовы» структуры характеризуются некоторыми универсальными или абсолютными свойствами по отношению к итеративным комплексным отображениям. Изучение таких структур является предметом отдельного раздела математики – так называемой теории комплексных динамических систем.

§3.7. Платоническая реальность математических понятий?

Насколько реальны объекты математического мира? Некоторые считают, что ничего реального в них быть не может. Математические объекты суть просто понятия, они представляют собой мысленные идеализации,[189] созданные математиками – часто под влиянием внешних проявлений и кажущегося порядка окружающего нас мира; но при этом они – всего лишь рожденные разумом абстракции. Могут ли они представлять собой что-либо, кроме просто произвольных конструкций, порожденных человеческим мышлением? И в то же время эти математические понятия часто выглядят глубоко реальными и эта реальность выходит далеко за пределы мыслительных процессов любого конкретного математика. Тут как будто имеет место обратное явление – человеческое мышление как бы само оказывается направляемым к некой внешней истине – истине, которая реальна сама по себе,[190] и которая открывается каждому из нас лишь частично.

Множество Мандельброта представляет собой потрясающий пример. Его удивительно сложная структура не является результатом изобретения ни какой-либо отдельной личности, ни группы математиков.[191] Сам Бенуа Мандельброт – американский математик польского происхождения (и один из главных разработчиков теории фракталов), который первый[192] изучил это множество, не мог себе представить, насколько фантастически сложным окажется этот объект, хотя и понимал, что обнаружил нечто очень интересное. Действительно, увидев самые первые компьютерные изображения, он счел увиденные им размытые структуры результатом сбоя (Мандельброт [1986])! И только потом он убедился, что они действительно являлись частью множества. Более того, сложную структуру множества Мандельброта во всех ее деталях не под силу охватить никому из нас, и ее невозможно полностью отобразить на компьютере. Создается впечатление, что рассматриваемая структура не является всего лишь частью нашего мышления, но что она реальна сама по себе. Кто бы из математиков или программистов ни занялся изучением этого множества, результатом их исследований обязательно будут приближения к

---

[189] **В.Э.:** «Мысленные идеализации»... Как беспомощно! И как же эти «мысленные идеализации» нам встроить в робота? (Ах это невозможно?)... На самом деле эти «мысленные идеализации» являются потенциальными продуктами программ, построенные бокоанализом другими программами, – и все эти программы (во всяком случае для достаточно квалифицированного программиста) являются объектами столь же определенными, как и операционные системы или вычислительные сети.

[190] **В.Э.:** Разумеется, она реальна! Если дана программа (или алгоритм), то ее продукты одинаковы для всех; это объективная данность – как и данности физического мира.

[191] **В.Э.:** Разумеется: множество Мандельброта – это потенциальный продукт описанного выше Пенроузом алгоритма. (А черные и белые точки на экране или бумаге в рисунках 3.1–3.7 – это даже и не потенциальный, а реализованный продукт этого алгоритма).

[192] Первенство обнаружения этого множества до сих пор остается предметом споров (см. Брукс, Мательски [1981], Мандельброт [1989]), но сама возможность таких споров представляет собой дополнительное свидетельство в пользу того, что здесь мы имеем дело скорее с открытием, чем с изобретением. **В.Э.:** Соотношения между открытием и изобретением здесь (как и повсюду в математике) очень просты: алгоритмы изобретают; их свойства потом открывают.

одной и той же единой для всех фундаментальной математической структуре. Не важно, на каком компьютере проводятся вычисления – лишь бы он правильно работал (конечно, если отвлечься от различий в степени подробности выявляемых деталей и скорости их вывода, связанными с различиями в производительности, объеме памяти и параметрах монитора). При этом компьютер применяется в сущности так же, как прибор в руках физика-экспериментатора, исследующего строение физического мира. Множество Мандельброта – это не плод человеческого воображения, а открытие. Подобно горе Эверест, множество Мандельброта просто-напросто уже существовало «там вовне»[193]!

Аналогичным образом сама система комплексных чисел обладает глубокой и вневременно́й реальностью, выходящей далеко за пределы мысленных конструкций, созданных любым конкретным человеком. Первые шаги на пути к пониманию комплексных чисел связаны с работами Джероламо Кардано. Он родился и жил в Италии с 1501 по 1576 год – врач, игрок и составитель гороскопов (однажды он даже составил гороскоп для Иисуса Христа), написавший в 1545 году очень важный и оказавший большое влияние на последующее развитие математики трактат по алгебре под названием *Ars Magna*. В этом трактате он предложил первое полное решение (в терминах иррациональных выражений, то есть корней *n*-ой степени) кубического уравнения в общем виде.[194] Кардано заметил, что в некоторых – так называемых «неприводимых» – случаях, когда уравнение имело три действительных решения, он был вынужден на определенном этапе включать в свою формулу квадратный корень из отрицательного числа. Хотя это обстоятельство и приводило его в замешательство, он понял, что полное решение можно получить тогда и только тогда, если допустить возможность извлечения таких квадратных корней (окончательный результат всегда оказывался действительным числом). Позднее, в 1572 году Рафаэль Бомбелли в своей работе, озаглавленной «Алгебра», обобщил работу Кардано, положив начало изучению алгебры комплексных чисел.

Хотя вначале может показаться, что введение таких квадратных корней из отрицательных чисел представляет собой всего лишь некоторый прием – математическое изобретение для достижения конкретной цели, – впоследствии становится очевидным, что потенциал этих объектов выходит далеко за рамки их использования для первоначально поставленных целей. При том, что изначально комплексные числа вводились (как уже упоминалось выше) для обеспечения возможности «безнаказанно» извлекать квадратные корни из отрицательных чисел, сделав этот шаг, мы получили в качестве бесплатного приложения еще и способ извлечения корней любой степени, а также решения любых алгебраических уравнений. Далее мы обнаружим у комплексных чисел много других волшебных свойств, о которых мы вначале даже и не подозревали. Эти свойства просто-напросто уже существуют «там вовне». Они не были привнесены туда ни Кардано, ни Бомбелли, ни Уоллисом, ни Котсом, ни Эйлером, ни Весселем, ни Гауссом, несмотря на несомненную прозорливость и их, и других великих математиков. Этот набор волшебных свойств был изначально присущ самой структуре, которую они шаг за шагом открывали. Когда Кардано вводил комплексные числа, он и подозревать не мог о существовании множества открытых впоследствии чудесных свойств, названных именами знаменитых ученых – таких как интегральная формула Коши, теорема отображения Римана или свойство продолжения Леви. Эти и многие другие замечательные свойства присущи самим числам – в точности тем самым числам, с которыми Кардано впервые столкнулся в 1539 году.

Что такое математика – изобретение или открытие? Процесс получения математиками результатов – что это: всего лишь построение не существующих в действительности сложных мысленных конструкций, мощь и элегантность которых способна обмануть даже их собственных изобретателей, заставив их поверить в «реальность» этих не более чем умозрительных построений? Или же математики действительно открывают истины уже где-то существующие, чья реальность в значительной степени независима от их деятельности? Я думаю, что читателю

---

[193] **В.Э.:** Множество Мандельброта «уже существовало там вовне» в том смысле, что ЕСЛИ кто-то миллион лет назад на Земле или кто-то на планете в Туманности Андромеды стал бы изучать сходимость в области комплексных (планарно ориентированных) чисел, то он получил бы тот же результат: в любой точке Вселенной (и также в других вселенных) в любое время как до Большого взрыва, так и после Большого коллапса. В этом смысле Платоновский мир (и множество Мандельброта в нем) вечен. Но другое дело, что могли бы и не изобрести такой алгоритм, основанный на сходимости в области комплексных чисел, и не изучать его. В этом смысле множество Мандельброта не существовало, пока им не занялись. (Не вижу тут ничего неясного).

[194] Частично основанное на более ранних работах Сципионе дель Ферро и Тартальи.

должно стать уже совершенно ясно, что я склонен придерживаться скорее второй, чем первой точки зрения, по крайней мере, в отношении таких структур, как комплексные числа или множество Мандельброта.[195]

Однако, не всё так просто. Как я уже сказал, в математике существуют вещи, к которым термин «открытие» подходит больше, чем «изобретение» – как в только что упомянутых примерах. Это происходит, когда структура дает гораздо больше того, что в нее было вложено изначально. Можно встать и на такую точку зрения, согласно которой в этих случаях математики просто наталкиваются на «творения Бога». Встречаются, однако, другие ситуации, когда математические структуры не столь убедительно уникальны – например, когда посреди доказательства какого-нибудь результата возникает необходимость в некой хитроумной, хотя и далеко не уникальной конструкции для достижения весьма специфической цели. В этих случаях от вновь созданной конструкции вряд ли следует ожидать больше того, что было в нее первоначально заложено, и термин «изобретение» представляется более подходящим, чем «открытие». Они, действительно, суть просто «творения человека». Согласно этой точке зрения, истинные математические открытия должны, как правило, рассматриваться как достижения более великие, чем «просто» изобретения.

Такого рода ранжирование обнаруживает некоторое сходство с тем, что мы иногда наблюдаем в области искусства или техники. Великие произведения искусства действительно «ближе к Богу», чем менее значительные творения. У художников нередко возникает чувство, что в своих величайших произведениях они открывают вечные истины, существовавшие уже до них в некотором высшем смысле,[196] в то время как менее значительные произведения могут быть более случайными, являясь по своей природе всего лишь порождениями простых смертных. Точно так же и новое инженерное решение с очень красивой структурой, позволяющее достичь значительных результатов через применение простой и неожиданной идеи, может с полным на то основанием рассматриваться скорее не как изобретение, а как открытие.

Однако, высказав все эти соображения, я не могу отделаться от ощущения, что в случае математики вера в некоторое высшее вечное существование – по крайней мере для наиболее глубоких математических концепций, – имеет под собой гораздо больше оснований, чем в других областях человеческой деятельности. Несомненная уникальность и универсальность такого рода математических идей по своей природе существенно отличается от всего того, с чем приходится сталкиваться в области искусства и техники. Точка зрения, согласно которой математические понятия могут существовать в такого рода вневременном, высшем смысле, была впервые высказана еще в древности (около 360 года до н.э.) великим греческим философом Платоном, и поэтому ее часто называют математическим платонизмом. Она играет важную роль в дальнейшем изложении.

В главе 1 я довольно много места уделил обсуждению точки зрения сильного искусственного интеллекта, согласно которой мыслительные явления находят свое воплощение в рамках математического понятия алгоритма. В главе 2 я особо подчеркнул, что алгоритм есть действительно очень глубокое и «Богом данное» понятие. В этой главе я старался доказать, что такие «Богом данные» математические идеи существуют в определенном смысле вне времени и независимо от нас смертных. Не могут ли эти соображения служить своего рода подтверждением справедливости концепции сильного искусственного интеллекта, допуская возможность некоего высшего существования мыслительной деятельности? Это вполне возможно – и я даже собираюсь далее привести ряд соображений в поддержку в чем-то похожей точки зрения. Но если у мыслительных явлений и вправду имеется такое вместилище, я всё же не думаю, что это может относиться и к понятию алгоритма. Тут нужно что-то более «тонкое». Последующее обсуждение будет в значительной степени опираться на тот факт, что связанные с понятием алгоритма объекты составляют очень узкую и ограниченную часть математики. Следующая глава даст некоторое представление об огромных возможностях и изяществе неалгоритмической математики.

---

[195] **В.Э.:** А если знать их сущность (что это: потенциальные продукты алгоритмов), то можно без всяких там «склонен», «не склонен» говорить смело и уверенно: «алгоритмы изобретают; их свойства открывают».

[196] Как сказал выдающийся аргентинский писатель Хорхе Луис Борхес: «...знаменитый поэт в большей степени первооткрыватель, чем изобретатель...».

Глава 4. Истина, доказательство и интуиция

§4.1. Программа Гильберта для математики

Что есть истина? Как мы составляем наши суждения о том, что в мире является справедливым, верным, а что – нет? Следуем ли мы некоторому алгоритму, которому отдается предпочтение среди прочих, менее эффективных, в процессе всемогущего естественного отбора? Или же возможен некий иной путь – не алгоритмизированный, а основанный на особой проницательности, интуитивный, инстинктивный – позволяющий угадывать правду? Это представляется нелегким вопросом.[197] Наши суждения зависят от сложных взаимосвязанных комбинаций данных, поставляемых органами чувств, и наших размышлений и догадок. Более того, во многих реальных ситуациях не может существовать единого мнения по поводу того, что на самом деле истинно, а что – ложно. Чтобы упростить задачу, рассмотрим только лишь математическую истину. Как мы формируем суждения – а может, даже и наши «стопроцентно верные» знания – при ответе на вопросы из области математики? Там уж, по крайней мере, всё должно быть не так размыто, очерчено более ясно. Там не может возникать вопросов об истинности – или все-таки может? Что же, в конце концов, есть математическая истина?

Вопрос об этой истине возник не сегодня, он уходит корнями в античность, к греческим философам и математикам – и, несомненно, еще дальше, в глубь веков. Однако, несколько великих открытий и поразительных прозрений здесь были сделаны не далее как в XX столетии. Эти новые достижения заслуживают того, чтобы постараться их понять. Они носят фундаментальный характер и непосредственно касаются вопроса о том, являются ли наши мыслительные процессы полностью алгоритмизированными по своей природе или нет. Четко разобраться в этом – задача, имеющая для нас весьма важное значение.

В последней части XIX века математика шагнула далеко вперед в результате развития всё более и более мощных методов математического доказательства. (Давид Гильберт и Георг Кантор, с которыми мы познакомились ранее, и великий французский математик Анри Пуанкаре, с которым нам еще предстоит встретиться, шли во главе этих разработок.) Как следствие, математики стали обретать уверенность в том, что применение этих методов приведет к успеху. Многие из таких методов основаны на рассмотрении множеств[198] с бесконечным числом членов, и доказательства часто оказывались осуществимы благодаря именно тому, что такое множество можно было рассматривать как реальный «объект» – завершенное единое целое, существующее не только в абстракции. Многие из этих идей родились из в высшей степени оригинальной концепции Кантора о бесконечных числах, которую он развил, последовательно используя бесконечные множества. (Мы кратко ознакомились с ними в предыдущей главе.)

Однако эта уверенность пошатнулась, когда в 1902 году английский логик и философ Бертран Рассел придумал свой знаменитый парадокс (который предвидел и сам Кантор и который выводился непосредственно из его диагонального процесса). Чтобы понять доводы Рассела, мы сначала должны хотя бы немного почувствовать, как можно представить множество в виде единого целого. Давайте представим себе множество, характеризуемое некоторым (общим) свойством.[199] Например, набор красных предметов может быть охарактеризован словом

---

[197] **В.Э.:** Если принят основной постулат Веданской теории (что человеческая интеллектуальная деятельность представляет собой обработку информации), то установление истины, разумеется, есть алгоритмический процесс. И именно потому, что разными субъектами могут применяться разные алгоритмы (и задействованные в них критерии), они приходят к разным выводам об истинности. А сущность «озарений» и «интуиции» была показана в {PENRO1 = с.88 здесь}.

[198] «Множество» означает набор предметов – физических объектов или математических абстракций, – который может рассматриваться как единое целое. В математике элементы (т.е. члены) множества часто сами являются множествами, поскольку множества могут собираться таким образом, чтобы самим формировать множества. Тем самым можно рассматривать множества множеств, множества множеств множеств и т.д.

[199] **В.Э.:** То, о чем Пенроуз сейчас говорит, соответствует реалии (см. Рис. VE1 в Предисловии этого тома). Собственно множество – это реалия. Но рассуждения о множестве не могут быть точными и исчерпывающими, если параллельно не помнить и о двух других объектах, сопутствующих реалии (множеству): о ее номиналии и о программе, определяющей множество. То «общее свойство, характеризующее множество», о котором сейчас говорит Пенроуз, – оно как раз и встроено в программу: именно

«краснота»[200] как его определяющим свойством: нечто принадлежит этому множеству тогда и только тогда, когда это обладает «краснотой» (имеет красный цвет). Это позволит нам «перевернуть» точку зрения и трактовать свойство как единичный объект, который будет состоять из всего множества вещей, обладающих данным свойством.[201] При таком рассмотрении «краснота» эквивалентна множеству всех красных предметов. (При этом мы можем предполагать существование «там вовне» и других множеств, члены которых не могут быть охарактеризованы подобным простым свойством.)

Идея формулировки понятий в терминах множеств послужила основой для процедуры, предложенной в 1884 году влиятельным немецким логиком Готтлибом[202] Фреге, которая позволяла определять числа через множества. К примеру, что мы понимаем под числом 3? Мы знаем, в чем заключается «тройственность», но что есть число 3 само по себе? Очевидно, что «тройственность» есть свойство наборов объектов, т.е. свойство множеств: некоторое множество обладает данным свойством тогда и только тогда, когда это множество состоит из трех членов. Этим свойством характеризуется, скажем, тройка призеров-медалистов некоторой Олимпиады. Равно как и набор шин к трехколесному велосипеду, или листья на одном стебельке обычного клевера, или множество всех решений уравнения $x^3 - 6x^2 + 11x - 6 = 0$. Как же можно тогда определить по Фреге само число 3? Согласно Фреге, 3 – это множество множеств, а именно, всех множеств, имеющих свойство «тройственности»[203]. Таким образом, множество содержит три члена тогда и только тогда, когда оно принадлежит множеству 3 по Фреге.

Может показаться, что мы попадаем в замкнутый круг, но в действительности это совсем не так. Мы можем определить числа в общем случае как совокупности всевозможных эквивалентных множеств, где говоря «эквивалентные», мы понимаем «состоящие из элементов, которые могут быть попарно сопоставлены друг другу» (или, в более привычной терминологии, «имеющих одинаковое число элементов»[204]). Тогда число 3 будет одной из этих совокупностей множеств,[205] которая содержит в себе в качестве члена множество, состоящее, скажем, из яблока, апельсина и груши. Обратите внимание, что это принципиально отличается от определения «**3**», данного Черчем (с. 69 {с.86 здесь}). Существуют также и другие определения, причем более популярные в наши дни.

Вернемся теперь к парадоксу Рассела. В чем он заключается? В нем рассматривается множество R, определенное следующим образом:

программа (ее алгоритм) определяет, что это за свойство и как отличить объекты, обладающие этим свойством, от объектов, им не обладающих.

[200] **В.Э.:** А точнее – не словом, а (мозговой) программой, способной установить, отражает ли данный объект свет таким образом, что поглощает фотоны всех других частот, кроме красного диапазона.

[201] **В.Э.:** Довольно туманное рассуждение... В оригинале: «*We may imagine some set that is characterized in terms of a particular PROPERTY. For example, the set of RED things is characterized in terms of the property of REDNESS: something belongs to that set if and only if it has redness. This allows us to turn things about, and talk about a property in terms of a single object, namely the entire set of things with that property. With this viewpoint, `redness' IS the set of all red things*.» Нет, – «краснота» не есть множество красных вещей; «краснота» есть способность отражать фотоны только красного спектра, а все остальные поглощать. И еще тут присутствует программа, алгоритм которой способен это свойство констатировать.

[202] **В.Э.:** Готлоб (в русских энциклопедиях) или хотя бы Готтлоб (как пишется по-немецки), но не Готтлиб – ошибка перевода: у Пенроуза правильно.

[203] Рассматривая множества, члены которых, в свою очередь, также являются множествами, мы должны тщательно проводить отличия между членами такого множества и членами его членов. Например, допустим, что S – это множество непустых подмножеств некоторого другого множества T, членами которого являются один апельсин и одно яблоко. T в таком случае имеет свойство «двойственности», тогда как S обладает свойством «тройственности»: членами S будут множества а) из одного яблока; б) из одного апельсина и в) из одного апельсина и одного яблока – представляющие три члена S. Аналогично, множество, чьим единственным членом является пустое множество, будет иметь свойство «единичности», а не «нулевости» – в него входит один член, а именно пустое множество! При этом само пустое множество не будет иметь, конечно, ни одного члена.

[204] **В.Э.:** Перевод точен; у Пенроуза тоже: «(i.e. in ordinary terms this would be `having the same number of members')». Но лучше было бы сказать не «имеющих одинаковое число элементов», а «имеющих одинаковое количество элементов», чтобы не употреблять понятие числа, когда оно еще не введено, а пока еще только вводится сейчас.

[205] **В.Э.:** Фреге не смог дойти до конца в понимании природы чисел, потому что не рассматривал всё это дело как работу компьютера. Разумеется, это простительно, если учесть то время, когда он жил (1848 – 1925). Компьютеров тогда не было, и опыта программирования Фреге не имел.

R есть множество множеств, которые не являются членами самих себя.[206]

Таким образом, *R* есть набор множеств *X*, отвечающих следующему условию: среди членов множества *X* не должно быть самого *X*.

Не является ли абсурдным предполагать, что множество в действительности может быть членом самого себя? Ничуть. Рассмотрим, к примеру, множество *I*, состоящее из бесконечных множеств (множеств с бесконечным числом членов). С очевидностью, существует бесконечное число различных бесконечных множеств, и само множество *I*, таким образом, является бесконечным. И, таким образом, оно, действительно, принадлежит самому себе! Но как же, в таком случае, рассуждения Рассела дают нам парадоксальное утверждение? Давайте спросим: является ли множество Рассела *R* членом самого себя или нет? Если нет, то оно должно принадлежать себе, ибо *R* состоит как раз из таких множеств, которые не являются членами самих себя. То есть, в конечном счете, *R* принадлежит *R* – противоречие! С другой стороны, если *R* есть член самого себя, то, поскольку «самое себя» – это *R*, оно в то же время принадлежит множеству, члены которого, по определению, не могут быть составляющими самих себя, т.е. все-таки не принадлежит самому себе – и вновь противоречие![207]

Этот парадоксальный вывод не был праздной игрой ума: Рассел использовал – хотя и в крайней форме – тот же тип весьма общих теоретико-множественных методов, которые математики начинали использовать в то время для своих доказательств. Становилось очевидным, что казавшаяся незыблемой почва ускользает из-под ног, и поэтому необходимо было как можно точнее определить, какие рассуждения считать допустимыми.[208] Ясно было, что такие рассуждения должны быть свободны от внутренних противоречий, и что утверждения, которые будут выводиться с их помощью как следствия из априори верных посылок, должны быть также верными. Рассел, совместно со своим коллегой Альфредом Нортом Уайтхедом, взялся за развитие такой полностью формализованной системы аксиом и правил вывода, на язык которой стало бы возможным перевести все виды корректных математических рассуждений. Все правила подвергались тщательному отбору, дабы избежать «ложных» путей рассуждений, могущих привести к парадоксам, подобным упомянутому выше. Однако схема, появившаяся на свет в результате этих усилий, была очень громоздка и оказалась весьма ограниченной по диапазону различных типов математических рассуждений, которые она охватывала. Великий математик Давид Гильберт (которого мы впервые встретили в главе 2 {с.55 здесь}) задался целью создать более практичную и универсальную систему. В нее должны были войти все типы математических рассуждений из всех областей математики. Более того, Гильберт стремился

---

[206] **В.Э.:** Ну вот, и чтобы разобрать (и понять) сущность знаменитого-презнаменитого «парадокса Рассела», обратимся опять к концепции множества, отображенной на Рис. VE1. Расселовское «множество *R*» – это реалия. Но рядом тут существует еще и номиналия и (главное!) программа. Вот эта программа-то и есть самое ключевое звено здесь, на что мы должны в первую очередь смотреть! Итак, Бертран Рассел на самом деле задал программу, которая будет проверять, принадлежит ли данное множество само себе в качестве элемента. (И, как обычно – как в случае с «краснотой» и «тройственностью» –, при наличии требуемого свойства причислять объект к задаваемому программой множеству-реалии). Ну, программа как программа – отчего же не придумать такой алгоритм? А теперь посмотрим (детально, а не поверхностно! посмотрим так, как смотрят программисты, а не так, как смотрят математики!), что же произойдет, если программу Рассела выполнить? Запускаем ее первый раз (фактически – над номиналией множества *R*, но одновременно и как бы над реалией – над самим множеством). Сейчас *R* не содержит себя в качестве элемента, значит причисляем его к *R*. Пускаем программу второй раз для проверки: теперь *R* уже принадлежит само себе, значит, исключаем его из *R*. Пускаем программу еще раз для надежности: теперь *R* не принадлежит само себе, значит, причисляем его к *R*. Пускаем программу для проверки... И так до бесконечности. Программа Рассела зациклилась! Ой, читатель, какой ужас, какой кошмар! Фреге чуть не застрелился и больше ничего не публикует всю оставшуюся жизнь! Крушение всех оснований математики! Мировая катастрофа, эхо которой вот уже 109 лет с 1902 года разносится по всему свету! Математики взмахивают руками, хватаются за головы и падают в обморок. Программа Рассела циклит!!! (Сравни с «парадоксом Лжеца» в {POTI-1 = МОИ № 41} стр.53).

[207] Можно дать занятную трактовку парадокса Рассела в более привычных терминах. Представьте себе библиотеку с двумя каталогами, один из которых перечисляет только те книги в библиотеке, которые хотя бы раз ссылаются на себя самих, а другой – остальные книги, т.е. те, которые не упоминают себя. В каком из этих каталогов, в таком случае, должен фигурировать второй каталог?

[208] **В.Э.:** Допустимыми следует считать лишь те рассуждения, которые учитывают тот факт, что всё это происходит в (мозговом) компьютере, а не невесть где. Тогда не только не будет «внутренних противоречий», а вопрос о противоречиях вообще станет столь же маловажным, как, например, в физике.

сделать возможным строгое доказательство отсутствия противоречий в своей схеме. Тогда математика раз и навсегда смогла бы встать на прочную и неколебимую основу.

Однако надежды Гильберта и его последователей были перечеркнуты, когда в 1931 году блестящий австрийский логик математики Курт Гёдель выдвинул поразительную теорему, которая до основания разрушала программу Гильберта. Гёдель показал, что любая подобная точная («формальная») система аксиом и правил вывода, если только она достаточна широка, чтобы содержать в себе описания простых арифметических теорем (как, например, «последняя теорема Ферма», рассмотренная в главе 2), и если она свободна от противоречий – то такая система должна включать утверждения, которые не являются ни доказуемыми, ни недоказуемыми в рамках формализма данной системы. Истинность таких «неразрешимых» утверждений, следовательно, не может быть выяснена с помощью методов, допускаемых самой системой. Более того, Гёдель смог показать, что даже утверждение о непротиворечивости системы аксиом, будучи переведенным в форму соответствующей теоремы, само по себе является «неразрешимым». Для нас будет очень важным понять природу этой неразрешимости. Тогда мы увидим, почему выводы Гёделя опровергали самое основание программы Гильберта. Мы также увидим, каким образом они дают нам возможность, воспользовавшись интуицией, выходить за пределы любой рассматриваемой формализованной математической системы. Это понимание будет решающим для того, чтобы, в свою очередь, лучше понять обсуждаемое далее.

§4.2. Формальные математические системы

Необходимо будет несколько уточнить, что мы понимаем под «формальными математическими системами аксиом и правил вывода». Мы должны предположить наличие некоторого алфавита символов, через которые будут записываться математические выражения. Эти символы в обязательном порядке должны быть адекватны для записи натуральных чисел с тем, чтобы в нашу систему могла быть включена «арифметика». По желанию, мы можем использовать общепринятую арабскую запись 0, 1, 2, 3, ..., 9, 10, 11, 12, ... хотя при этом конкретные выражения для правил вывода становятся несколько более сложными, чем требуется. Гораздо более простые выражения получаются, скажем, при использовании записи вида 0, 01, 011, 0111, 01111, ... для обозначения последовательности натуральных чисел (или, в качестве компромисса, мы могли бы использовать двоичную запись). Однако, поскольку это могло бы стать источником разночтений в дальнейших рассуждениях, я буду для простоты придерживаться обычной арабской записи независимо от способа обозначения, которая может на самом деле использоваться в данной системе. Нам мог бы понадобиться символ «пробел» для разделения различных «слов» или «чисел» в нашей системе, но, так как это тоже может вызвать путаницу, то мы будем по мере необходимости использовать для этих целей просто запятую (,). Произвольные («переменные») натуральные числа (равно как и целые, рациональные и т.д.; но давайте здесь ограничимся натуральными) мы станем обозначать буквами, например, $t, u, v, w, x, y, z, t', t'', t'''$ и т.п. Штрихованные буквы t', t'', ... вводятся нами в употребление, дабы не ограничивать число переменных, которые могут встретиться в произвольном выражении. Мы будем считать штрих (′) отдельным символом формальной системы, так что действительное количество символов в системе остается конечным. Помимо этого нам также потребуются символы для базовых арифметических операций –, +, × и т.д.; для различных видов скобок (,), [,], и для обозначения логических операций, таких как & («и»), ⇒ («следует»), ∨ («или»), ⇔ («тогда и только тогда»), ~ («не»). Дополнительно нам будут нужны еще и логические «кванторы»: квантор существования ∃ («существует... такое, что») и квантор общности ∀ («для любого... выполняется»). Тогда мы сможем такие утверждения, как, например, «последняя теорема Ферма», привести к виду:

$$\sim\exists\, w, x, y, z\, [(x + 1)^{w+3} + (y + 1)^{w+3} = (z + 1)^{w+3}]$$

(см. главу 2, с. 62 {с.76 здесь}). (Я мог бы написать «0111» для «3», и, возможно, использовать для «возведения в степень» обозначение, более подходящее к рассматриваемому формализму; но, как уже говорилось, я буду придерживаться стандартной системы записи во избежание ненужной путаницы.) Это утверждение (если читать его до левой квадратной скобки) звучит как:

«Не существует таких натуральных чисел w, x, y, z, что...».

Мы можем также переписать последнюю теорему Ферма при помощи ∀:

$$\forall\, w, x, y, z\, [\sim (x + 1)^{w+3} + (y + 1)^{w+3} = (z + 1)^{w+3}]$$

которое будет читаться следующим образом (заканчивая символом «не» после левой квадратной скобки):

«Для любых натуральных чисел w, x, y, z не может быть выполнено...»,
что логически эквивалентно написанному ранее.

Нам понадобятся еще и буквы, обозначающие целые утверждения, для чего я буду использовать заглавные буквы $P, Q, R, S, \ldots$. Таким утверждением может, к примеру, служить и вышеприведенная теорема Ферма:

$$F = \sim\exists\, w, x, y, z\, [(x + 1)^{w+3} + (y + 1)^{w+3} = (z + 1)^{w+3}]$$

Утверждение может также зависеть от одной или более переменных; например, нас может интересовать формулировка теоремы Ферма для некоторого конкретного[209] значения степени w+3

$$G(w) = \sim\exists\, x, y, z\, [(x + 1)^{w+3} + (y + 1)^{w+3} = (z + 1)^{w+3}]$$

так что $G(0)$ утверждает, что «куб не может быть суммой кубов положительных чисел»; $G(1)$ говорит о том же применительно к четвертым степеням и так далее. (Обратите внимание на отсутствие w после символа $\exists$.) Тогда теорема Ферма гласит, что $G(w)$ выполняется для любого w:

$$F = \forall w[G(w)].$$

G( ) является примером так называемой функции исчисления высказываний, т.е. утверждением, которое зависит от одной или более переменных.

Аксиомы нашей системы будут представлять из себя перечень утверждений общего характера, чья справедливость в рамках принятого символизма предполагается самоочевидной. Например, для произвольных утверждений или функций исчисления высказываний P, Q, $R($ $)$ мы могли бы указать среди прочих аксиом системы такие, как

$$(P \,\&\, Q) \Rightarrow P,$$
$$\sim(\sim P) \Leftrightarrow P,$$
$$\sim\exists\, x\, [R(x)] \Leftrightarrow \forall\, x\, [R(x)]$$

«априорная истинность» которых уже заключена в их смысловых значениях. (Первое утверждение означает лишь, что «если выполняется P и Q, то выполняется и P»; второе устанавливает равносильность утверждений «неверно, что не выполняется P» и «P выполняется»; а третье может быть проиллюстрировано эквивалентностью двух способов формулировки теоремы Ферма, данных выше.) Мы можем также включить основные аксиомы арифметики:

$$\forall\, x, y\, [(x + y) = (y + x)],$$
$$\forall\, x, y, z\, [(x + y) \times z = [(x \times z) + (y \times z)],$$

хотя некоторые предпочитают определять арифметические операции через более простые понятия и выводить вышеуказанные утверждения как теоремы. Правила вывода могут вводиться в виде (самоочевидных) процедур типа

«Из P и $P \Rightarrow Q$ следует Q».

«Из $\forall\, x\, [R(x)]$ мы можем вывести любое утверждение, получающееся путем подстановки конкретного натурального числа x в $R(x)$».

Такие правила являются инструкциями, следуя которым, можно с помощью утверждений, чья истинность уже доказана, получать новые утверждения.

Теперь, отталкиваясь от системы аксиом и раз за разом применяя правила вывода, мы имеем возможность построить достаточно длинные цепочки новых утверждений. На любой стадии этого процесса мы можем использовать снова и снова любую из аксиом, а также обратиться к любому из уже выведенных нами производных утверждений. Каждое утверждение из корректно выстроенной цепочки называется теоремой (несмотря на то, что многие из них достаточно тривиальны и неинтересны с точки зрения математики). Если у нас есть некое утверждение P, которое мы хотим доказать, то мы должны подобрать такую цепочку, выстроенную в согласии с действующими правилами вывода, которая заканчивается утверждением P. Такая цепочка предоставит нам доказательство P в рамках системы; а P тогда будет являться, соответственно, теоремой.

---

[209] Хотя справедливость теоремы Ферма в общем случае пока не доказана, ее справедливость для некоторых частных случаев, таких как $G(0)$, $G(1)$, $G(2)$, $G(3)$, доказана вплоть до $G(125000)$. Другими словами, доказано, что куб никакого числа не может быть суммой кубов двух других положительных чисел, четвертая степень числа не может быть суммой четвертых степеней других чисел и т.д. вплоть до степени 125˙000. (Несколько лет назад теорема Ферма была доказана в общем виде. См. сноску 5 на с. 62. – *Прим. ред*.)

Идея программы Гильберта состояла в том, чтобы найти применительно к любой отдельно взятой области математики набор аксиом и правил вывода, который был бы достаточно полным для всех возможных в данной области корректных математических рассуждений. Пусть такой областью будет арифметика (с добавленными кванторами ∃ и ∀, позволяющими формулировать утверждения, подобные последней теореме Ферма). То, что мы не рассматриваем более общую область математики, не умаляет нашу задачу: арифметика и сама по себе обладает общностью, достаточной для применения процедуры Гёделя. Если мы допустим, что благодаря программе Гильберта мы действительно располагаем такой всеобъемлющей системой аксиом и правил вывода для арифметики, то мы тем самым обретаем и определенный критерий для выявления «корректности» математического доказательства любого утверждения в области арифметики. Возлагались надежды на то, что подобная система аксиом и правил может быть полной в смысле предоставляемой нам принципиальной возможности решать, истинно или ложно произвольное утверждение, сформулированное в рамках этой системы.

Гильберт рассчитывал, что для любой строки символов, представляющих математическое утверждение, скажем, *P*, можно будет доказать либо *P*, либо ~*P*, если *P* истинно или ложно, соответственно. Здесь мы в обязательном порядке оговариваем, что строка должна быть синтаксически корректна, где «синтаксически корректна» по сути означает «грамматически корректна» – то есть удовлетворяет всем правилам записи, принятым в данном формализме, среди которых будет правильное попарное соответствие скобок и т.п. – так чтобы *P* всегда имело четко определенное значение «ложь» или «истина». Если бы надежды Гильберта оправдались, то можно было бы вообще не задумываться о том, что означает то или иное утверждение! *P* было бы просто-напросто синтаксически корректной строкой символов. Строке было бы приписано значение ИСТИНА, если бы *P* являлось теоремой (другими словами, если бы *P* было доказуемо в рамках системы); или же ЛОЖЬ, если бы теоремой было ~*P*. Чтобы такой подход имел смысл, мы должны дополнительно к условию полноты наложить еще и условие непротиворечивости, гарантирующее отсутствие такой строки символов *P*, для которой как *P*, так и ~*P* были бы теоремами. Ведь в противном случае *P* могло бы быть одновременно и ИСТИНОЙ, и ЛОЖЬЮ!

Такой подход, согласно которому можно пренебрегать смысловыми значениями математических выражений и рассматривать их лишь как строки символов некоторой формальной математической системы, в математике получил название формализма.[210] Некоторым нравится эта точка зрения, с которой математика превращается в своего рода «бессмысленную игру». Однако я сам не являюсь сторонником таких идей. Все-таки именно «смысл» – а не слепые алгоритмические вычисления – составляет сущность математики.[211] К счастью, Гёдель нанес формализму сокрушающий удар! Давайте посмотрим, как он это сделал.

---

[210] **В.Э.:** С точки зрения Веданской теории формализация является фундаментально неправильным и глубоко ошибочным направлением деятельности. Математика начиналась с первичных действий (над множествами), но расцвет свой достигла во вторичных действиях (над нотатами), которые сохраняют изоморфизм с первичными действиями и поэтому могут быть полезны. Формализация же открывает новый, третичный слой действий, опять же претендующий на некоторый изоморфизм со вторичными. Однако не нужно и не следует запутывать дело еще и третичным слоем, всё дальше уходящим от первичных начал математики; он не дает и не может дать ничего ценного. Правильные действия состоят, наоборот, в максимальном раскрытии и осознании отношений первичного и вторичного слоев математики. А (в общем-то похвальное) стремление как можно более точно осознать и фиксировать принципы мышления нужно реализовывать не в третичном слое, вдалеке от (настоящих) оснований математики, а в первичном слое непосредственно у ее истоков. С этой целью нужно осознать и строго описать те (мозговые) программы, которые на самом деле порождают математику. Для строгого описания программ в информатике придуманы алгоритмические языки. Следовательно (вместо формализации!) должны быть созданы алгоритмические языки для описания (и строгого фиксирования) оснований математики. (А для проверки, демонстрации и эмуляции – созданы компьютерные интерпретаторы этих языков). Такой подход в свое время (в 1984 г.) был назван мною «компьютерной канонизацией» {CANTO2.1253 = МОИ № 39} – в противоположность «математической формализации». Была также (в 1980 г.) предпринята попытка разработки такого алгоритмического языка под названием «Эуклидол» и интерпретатора под названием «Эуклидос» (см. {NATUR2 = МОИ № 35}).

[211] **В.Э.:** Здесь у Пенроуза всё оказалось перевернуто «вверх ногами». Формализацию следует отвергнуть именно потому, что она представляет собой уход от (первичных) алгоритмов математики. (Можно сказать, что она представляет собой замену настоящих алгоритмов математики другими – третичными, «шутовскими»). Нужно именно разбирать настоящие алгоритмы математики (первичные, а

§4.3. Теорема Гёделя

Часть доказательства, приведенного Гёделем, содержало некий очень сложный и детализированный кусок. Однако нам не обязательно разбираться во всех его тонкостях. Основная идея, в то же время, была проста, красива и глубока. И ее мы сможем оценить по достоинству. В «сложной» части (которая, впрочем, содержит много остроумных рассуждений) подробно показано, каким образом частные правила вывода и использование различных аксиом формальной процедуры могут быть представлены в виде арифметических операций. (Хотя в сложной части становится понятной плодотворность этих действий!) Для этого представления нам необходимо будет найти какой-нибудь удобный способ нумерации утверждений при помощи натуральных чисел. Один из способов мог бы заключаться в том, чтобы использовать своего рода «алфавитный» порядок для строчек символов формальной системы, имеющих одинаковую длину, упорядочить заранее строчки по длине. (Таким образом, за выстроенными в алфавитном порядке строками из одного символа будут следовать строки длиной в два символа, также упорядоченные по алфавиту; за ними идут строки из трех символов и так далее.[212]) Это называется лексикографическим порядком.[213] В действительности Гёдель использовал более сложную систему нумерации, но различия в данном случае для нас несущественны. Нас же должны в особенности интересовать функции исчисления высказываний одной переменной, наподобие введенной выше $G(w)$. Пусть n-я (из пронумерованных выбранным способом строк символов) такая функция от аргумента w обозначается[214]

$$P_n(w).$$

Мы можем допустить, чтобы наша нумерация по желанию была несколько «либеральна» в отношении синтаксически некорректных выражений. (Это позволит значительно упростить перевод системы на язык арифметических операций[215] по сравнению со случаем, когда мы будем стараться исключить из рассмотрения синтаксически некорректные выражения.) Если $P_n(w)$ синтаксически корректно, то оно будет представлять из себя некоторое совершенно определенное арифметическое выражение, в котором фигурируют два натуральных числа n и w.[216] Каков будет конкретный вид этого выражения – зависит от особенностей системы нумерации, которую мы выбрали. Но эти детали рассматриваются в «сложной» части и сейчас нас не касаются. Пусть

$$\Pi_n$$

будет n-м доказательством.[217] (Опять же мы можем использовать «либеральную нумерацию», когда для некоторых значений n выражение[218] Π_n не является синтаксически корректным и, тем самым, не доказывает никакую теорему.)

также вторичные). Но у Пенроуза понятие «алгоритма» оказалось связанным только с третичными, только с формализацией, только с этой «шелухой» по выражению Мориса Клайна {CANTO2.1505 = МОИ № 39}.

[212] **В.Э.:** Таким образом, если в «алфавите» s символов, то нумеруемый список состоит из $s + s^2 + s^3 + ... + s^z$ строк, где z – максимальная длина строки.

[213] Мы можем представить себе лексиграфический способ упорядочивания как обычный способ, используемый для натуральных чисел, только сделанный «по основанию $k + 1$», где для $k + 1$ чисел берутся различные символы формальной системы, вместе с новым «нулем», который никогда не используется. (Последняя сложность возникает в связи с тем, что числа, начинающиеся с нуля, и те, где он опущен – равны.) Простое лексикографическое упорядочивание в строчках из девяти символов осуществляется при помощи натуральных чисел, которые могут быть выписаны в стандартной десятичной системе без нуля: 1, 2, 3, 4, ..., 8, 9, 11, 12, ..., 19, 21, 22, ..., 99, 111, 112, ...

[214] **В.Э.:** Я как-то не очень понимаю: номер n – это в том же списке, где были все возможные строки и который состоял из $s + s^2 + s^3 + ... + s^z$ строк, или речь идет уже о другом списке, в котором находятся только лишь функции одной переменной типа $G(w)$?

[215] **В.Э.:** Я также не понимаю, какого черта сюда вообще нужно впутывать арифметику?

[216] **В.Э.:** Ну хорошо: $P_n(w)$, значит, есть некоторое формальное выражение; оно имеет номер n в некотором списке A (правда, не очень ясно, в каком) и что-то утверждает о числе w.

[217] **В.Э.:** «Доказательство» в формальной системе – это цепочка строк, по «правилам вывода» переходящих одна в другую. Так – стало быть, теперь мы (в другом списке B) перенумеровали все возможные цепочки строк. Сколько же их должно быть? Трудно даже сообразить... Тут пошла уже какая-то комбинаторика. Ясно во всяком случае, что комбинаций строк должно быть значительно больше, чем самих строк.

[218] **В.Э.:** Почему выражение?! Откуда взялось выражение!? Только что Π_n было названо доказательством! Доказательство – это цепочка выражений, ведущих к какому-нибудь одному выражению (доказанной теореме). Выражение – это одна строчка символов языка. У них что? – что цепочка строк, что

А теперь рассмотрим следующую функцию исчисления высказываний от натурального числа *w*:

$$\sim\exists\, x\, [\Pi_x \text{ доказывает } P_w(w)].$$

В выражении в квадратных скобках частично присутствуют слова, но, тем не менее, это – абсолютно точно определенное выражение. Оно говорит о том, что доказательство номер *x* является доказательством утверждения $P_w(\ )$, примененного к самому *w*. Находящийся за скобками квантор существования с отрицанием позволяет исключить из рассмотрения одну из переменных («не существует такого *x*, что...»), приводя нас в конечном счете к арифметической функции исчисления высказываний, зависящей только от *w*. В целом данное выражение утверждает, что не существует доказательства $P_w(w)$. Я буду предполагать, что оно оформлено синтаксически корректным образом (даже если $P_n(w)$ некорректно – поскольку тогда выражение было бы истинным за невозможностью существования доказательства синтаксически некорректного утверждения). На самом деле, в результате сделанного нами перевода на язык арифметики, написанное выше будет в действительности неким арифметическим выражением, включающим натуральное число *w* (тогда как в квадратных скобках окажется четко определенное арифметическое выражение, связывающее два натуральных числа *x* и *w*). Конечно, возможность представления этого выражения в арифметическом виде далеко не очевидна, но она существует. Рассуждения, приводящие к этому заключению, составляют наиболее трудную задачу в «сложной» части доказательства Гёделя. Как и ранее, непосредственный вид арифметического выражения будет зависеть от способа нумерации и в еще большей степени от конкретной структуры аксиом и правил вывода, принятых в нашей системе. Поскольку всё это входит в «сложную» часть доказательства, то в данном случае нас не интересует.

Мы пронумеровали все функции исчисления высказываний, зависящие от одной переменной, поэтому той, которую мы ввели выше, также должен быть приписан номер. Пусть этот номер будет *k*. Наша функция будет в таком случае *k*-й в общем списке. То есть

$$\sim\exists\, x\, [\Pi_x \text{ доказывает } P_w(w)] = P_k(w).$$

Теперь исследуем эту функцию при определенном значении: $w = k$. Мы получаем:

$$\exists\, x\, [\Pi_x \text{ доказывает } P_k(k)] = P_k(k).$$

Данное утверждение $P_k(k)$ является абсолютно точно определенным (синтаксически корректным) арифметическим выражением. Может ли оно быть доказано в рамках нашей формальной системы? А его отрицание $\sim P_k(k)$ – имеет ли оно такое доказательство? Ответ в обоих случаях будет отрицательный. Мы можем убедиться в этом путем исследования смысла, который лежит в основании процедуры Гёделя. Хотя $P_k(k)$ является просто арифметическим выражением, последнее было построено нами таким образом, что написанное в левой части утверждает следующее: «внутри системы не существует доказательства $P_k(k)$». Если мы были аккуратны в определении аксиом и процедур вывода, и не ошиблись при нумерации, то тогда в рамках системы такого доказательства найти невозможно. Если же доказательство существует, то значение утверждения, содержащегося в $P_k(k)$ – о том, что такого доказательства нет, – будет ложным, а вместе с ним будет ложным и арифметическое выражение, отвечающее $P_k(k)$. Но наша формальная система не может быть построена настолько плохо, чтобы включать в себя ложные утверждения, которые могут быть доказаны! Таким образом, в действительности, доказательство $P_k(k)$ быть не может. Но это в точности то самое, о чем говорит нам $P_k(k)$. То, что утверждает $P_k(k)$, обязано, следовательно, быть верным, а поэтому $P_k(k)$ должно быть верным как арифметическое выражение. Значит, мы нашли истинное утверждение, которое недоказуемо в рамках системы!

А как насчет $\sim P_k(k)$? Из предыдущих рассуждений видно, что доказательство этому утверждению внутри системы мы найти не сможем. Мы только что установили, что $\sim P_k(k)$ должно быть ложным (ибо $P_k(k)$ является истинным), а мы, по определению, не имеем возмож-

---

одна строчка – одно и то же и никакой разницы? (Всё это делает данное рассуждение Пенроуза бессмысленным набором слов и знаков, в котором бесполезно пытаться разобраться. А в других книгах изложение доказательства «теоремы Гёделя» не лучше. Я, к сожалению, не располагаю связным и осмысленным изложением, по которому была бы ясна сущность происходящего. Если они сами эту сущность понимают – в чем я, впрочем, сильно сомневаюсь, – то они совершенно не в состоянии это толково и непротиворечиво изложить на бумаге).

ности доказывать ложные утверждения в рамках системы! Таким образом, ни $P_k(k)$, ни $\sim P_k(k)$ недоказуемы в нашей формальной системе, что и составляет теорему Гёделя.[219]

§4.4. Математическая интуиция

Обратите внимание, что мы здесь сталкиваемся с одной примечательной особенностью. Часто думают, что теорема Гёделя имеет, в некотором роде, отрицательный смысл, поскольку она указывает на принципиальные ограничения в применении формальных математических рассуждений. Независимо от нашего мнения об универсальности применяемого подхода, всегда найдутся утверждения, которые не попадают в сферу его действия. Но насколько, в действительности, нас могут затрагивать частные случаи типа $P_k(k)$? В ходе предыдущих рассуждений мы установили, что $P_k(k)$ – истинное утверждение! Мы смогли это сделать несмотря на то, что это утверждение формально недоказуемо в рамках системы. А вот математических формалистов это должно волновать, потому что наши рассуждения с необходимостью приводят к выводам о неполноте их понятия «истины». Какая бы (непротиворечивая) формальная система не использовалась для арифметики, в ней будут содержаться утверждения, понимаемые нами как истинные, но которым не может быть приписано значение ИСТИНА при помощи вышеописанной формальной процедуры. Способ, при помощи которого формалист сумел бы обойти подобные трудности, мог бы состоять в том, чтобы не говорить о понятии истины, а только лишь о доказуемости внутри конкретной формальной системы. Однако же, такой подход весьма ограничен. Он не позволил бы даже сформулировать утверждение Гёделя и осуществить его доказательство, как это было сделано выше, поскольку в значительной части рассуждений речь идет как раз об определении того, что есть ложь, а что – истина.[220] Некоторые формалисты встают на более «прагматическую» точку зрения, заявляя, что их не волнуют утверждения, подобные $P_k(k)$, поскольку они исключительно сложны и не интересны в качестве арифметических выражений. Отстаивают они свою точку зрения примерно так:

> «Да, есть странные утверждения, вроде $P_k(k)$, для которых мое понятие доказуемости или ИСТИНЫ расходится с вашим интуитивным понятием истинности, но подобные выражения едва ли встречаются в серьезной математике (по крайней мере не в такой, которая меня интересует), поскольку они абсурдно усложнены и неестественны для математики».

Несомненно, что утверждения вида $P_k(k)$, будучи полностью выписанными, были бы чрезвычайно громоздки и выглядели бы странно для числовых математических выражений. Однако за последнее время были выдвинуты сравнительно простые выражения приемлемого с

---

[219] **В.Э.:** К сожалению, из-за нечеткого изложения Пенроузом (а также другими доступными мне авторами) исходных установок и хода рассуждений при доказательстве «теоремы Гёделя», я не могу в деталях проследить происходящее и проанализировать всё так, как сделал это с «теоремой Гёделя–Тьюринга» в {PENRS1}, с «теоремой Тьюринга» в {PENRO1 = с.82 здесь} и с «теоремой Кантора» в §3.3 выше в этом томе. Но происходящее здесь, конечно, доверия не внушает. Чувствуется тот же почерк, тот же стиль, те же методы. Опять ищется (скорее всего – несуществующее) пересечение $P_k(k)$ диагонали и строки; опять игнорируются всякие соображения о величине бесконечных множеств и над всем довлеет несокрушимый постулат $\infty/\infty \equiv 1$. Если кто-нибудь из читателей в дальнейшей переписке ответит на мои вопросы, поставленные в предыдущих сносках и тем самым объяснит, о чем, собственно, Пенроуз здесь говорит, то мы сможем проанализировать доказательство «теоремы Гёделя» до конца и оценить достоверность этого доказательства. А пока что я ограничусь лишь общей оценкой ситуации. И она заключается в следующем. Для Веданской теории не имеет никакого значения, верно или не верно доказательство «теоремы Гёделя». В любом случае это относится к третичному построению – околоматематическому (но не принадлежащему собственно математике!) и лишнему по всей своей сути. Если «теорема Гёделя» верна, то она разрушает это околоматематическое построение (и слава Богу!), но если она неверна, то всё равно ясно, что это построение никуда не годится. Никакого «вселенского» значения «теорема Гёделя» в любом случае не имеет, и ничего не говорит она ни о силе вообще логики, ни о применимости алгоритмов в мышлении или о сводимости этого мышления к алгоритмам. Там есть свои законы, свои выводы – и никакая теорема Гёделя там не нужна.

[220] В действительности ход рассуждений в теореме Гёделя может быть представлен таким образом, чтобы не зависеть от полностью привнесенного извне понятия «истины» для утверждений, подобных $P_k(k)$. Однако, он по-прежнему будет зависеть от интерпретации фактического «значения» некоторых символов: в частности, «$\sim\exists$» должно означать «не существует (натурального числа) ...такого, что...».

точки зрения математики характера, которые эквивалентны утверждениям Гёделя.[221] Они недоказуемы на основании обычных аксиом арифметики, однако же следуют из некоего свойства «самоочевидности», которым обладает сама система аксиом.

Отсутствие интереса к «математической истине», исповедуемое формалистами, кажется мне очень странной позицией в приложении к философии математики. Более того: она совсем не так прагматична, как представляется. Когда математики проводят свои выкладки, они не намерены постоянно проверять, могут ли они быть сформулированы посредством аксиом и правил вывода некоторой сложной формальной системы. Единственно, что необходимо – быть уверенным в правомерности использования этих рассуждений для установления истины. Доказательство Гёделя удовлетворяет этому требованию, так что $P_k(k)$ является математической истиной с таким же правом, как и любое другое утверждение, полученное более стандартным путем с использованием изначально заданных аксиом и правил вывода.

Процедура, которая напрашивается сама собой, заключается в следующем. Давайте положим, что $P_k(k)$ – совершенно верное утверждение (переобозначим его здесь как G_0). Тогда мы можем присоединить его к нашей системе в качестве дополнительной аксиомы. Естественно, что наша новая система будет, в свою очередь, содержать <u>новое</u> утверждение Гёделя, скажем, G_1, которое также будет истинным числовым выражением. Соответственно, мы можем и G_1 добавить в нашу систему. Это даст нам новую улучшенную систему, которая также содержит новое утверждение Гёделя G_2 (опять же совершенно справедливое); и мы сможем снова добавить его к системе, получая следующее утверждение Гёделя G_3, которое мы тоже присоединяем – и так далее, повторяя этот процесс неограниченно. Что мы можем сказать о получившейся в результате системе, где мы используем <u>весь</u> набор G_0, G_1, G_2, G_3, ... как дополнительные аксиомы? Может ли <u>эта</u> система быть полной? Поскольку мы теперь имеем неограниченную (бесконечную) систему аксиом, то возможность применения процедуры Гёделя совсем не очевидна. Однако, это последовательное включение утверждений Гёделя является в высшей степени систематичной схемой, результат применения которой может быть истолкован как обычная конечная система аксиом и правил вывода. Эта система будет иметь свое собственное утверждение Гёделя G_w, которое мы также сможем к ней присоединить, получая новую систему и с ней – еще одно утверждение Гёделя G_{w+1}. Продолжая, как и ранее, мы получаем набор утверждений G_w, G_{w+1}, G_{w+2}, G_{w+3}, ... , каждое из которых истинно и может быть включено в нашу формальную систему. Сохраняя свойство строгой систематичности, этот процесс вновь приводит нас к созданию новой системы, которая охватывает все созданные к этому моменту аксиомы. Но и эта система, в свою очередь, имеет свое собственное утверждение Гёделя, скажем, G_{w+w} – которое можно переписать как G_{w2}, и мы можем начать всю процедуру заново. В результате этого мы получим новый бесконечный, но систематический, набор аксиом G_{w2}, G_{w2+1}, G_{w2+2}, и т.д., приводящий к еще одной новой системе – и новому утверждению Гёделя G_{w3}. Воспроизводя весь процесс, мы получаем G_{w4}, потом – G_{w5} и так далее. И эта схема также будет полностью систематичной и даст свое собственное утверждение Гёделя G_{w^2}.

Есть ли логическое завершение у этого процесса? В определенном смысле – нет; но это приводит нас к ряду трудных математических рассуждений, которые здесь не могут быть нами рассмотрены во всех деталях. Вышеуказанная процедура обсуждалась Аланом Тьюрингом в статье,[222] опубликованной в 1939 году. Примечательно, что на самом деле <u>любое истинное</u> (в

---

[221] В нижеследующем прописные буквы будут представлять натуральные числа, а заглавные – конечные множества натуральных чисел. Пусть $m \to [n, k, r]$ представляет такое утверждение: «Если $X = \{0, 1, ..., m\}$, каждое из подмножеств которого длиной в k элементов приписано к r ящикам, то существует “большое” подмножество Y, принадлежащее X и имеющее по крайней мере n элементов, такое, что все подмножества Y из k элементов попадут в один ящик». Здесь «большое» означает, что число элементов, входящих в Y, больше самого маленького из натуральных чисел, принадлежащих Y. Рассмотрим теперь следующее утверждение: «При любых k, r, n существует m_0 такое, что при $m > m_0$ утверждение $m \to [n, k, r]$ всегда справедливо». Дж. Парисом и Л. Харрингтоном [1977] было доказано, что это положение эквивалентно гёделевскому утверждению для стандартных (введенных Пеано) аксиом арифметики, которое не выводится из этих аксиом и которое позволяет делать утверждения о тех аксиомах, которые «очевидно верны» (в данном случае оно говорит, например, о том, что утверждения, выведенные из аксиом, сами будут справедливыми).

[222] Статья называлась «Система логики, основанная на порядковых числах», и некоторые читатели будут уже знакомы со способом записи Канторовых <u>порядковых чисел</u>, который я применял для субиндексов. Иерархия логических систем, которые получаются с помощью приведенной мной процедуры,

общепринятом смысле) утверждение в арифметике может быть получено путем повторения процедуры «гёделизации» такого рода (см. Феферман [1988]). Однако это может вызвать вопрос о том, как мы в действительности решаем, является ли утверждение истинным или ложным. Исключительно важным будет также понять, как на каждом этапе нужно выполнять присоединение бесконечного семейства утверждений Гёделя, чтобы они порождали единственную дополнительную аксиому (или конечное число аксиом). Для выполнения такого присоединения требуется определенная алгоритмическая систематизация нашего бесконечного семейства. Чтобы быть уверенным в том, что подобная систематизация корректна и приводит к желаемому результату, нам придется опереться на интуитивные представления, выходящие за рамки системы – точь-в-точь, как мы это сделали для установления истинности $P_k(k)$. Именно эти «прозрения» и не могут быть систематизированы, не говоря о том, что они должны лежать вне сферы действия любой алгоритмической процедуры!

Интуитивная догадка, которая позволила нам установить, что утверждение Гёделя $P_k(k)$ является на самом деле истинным, представляет собой разновидность общей процедуры, известной логикам как принцип рефлексии: посредством нее, размышляя над смыслом системы аксиом и правил вывода и убеждаясь в их способности приводить к математическим истинам, можно преобразовывать интуитивные представления в новые математические выражения, невыводимые из тех самых аксиом и правил вывода. То, как нами была выше установлена истинность $P_k(k)$, как раз базировалось на применении этого принципа. Другой принцип рефлексии, имеющий отношение к доказательству Гёделя (хотя и не упомянутый выше), опирается на вывод новых математических истин исходя из представления о том, что система аксиом, которую мы полагаем априори адекватной для получения математических истин, является непротиворечивой. Применение принципов рефлексии часто подразумевает размышления о бесконечных множествах, и при этом нужно быть всегда внимательным и остерегаться рассуждений, которые могут привести к парадоксам наподобие расселовского. Принципы рефлексии полностью противопоставляются рассуждениям формалистов. Если использовать их аккуратно, то они позволяют вырваться за жесткие рамки любой формальной системы и получить новые, основанные на интуитивных догадках, представления, которые ранее казались недостижимыми. В математической литературе могло бы быть множество приемлемых результатов, чье доказательство требует «прозрений», далеко выходящих за рамки исходных правил и аксиом стандартной формальной системы арифметики. Всё это свидетельствует о том, что деятельность ума, приводящая математиков к суждениям об истине, не опирается непосредственно на некоторую определенную формальную систему.[223] Мы убедились в истинности утверждения Гёделя $P_k(k)$, хотя мы и не можем вывести ее из аксиом системы. Этот тип «видения», используемый в принципе рефлексии, требует математической интуиции, которая не является результатом чисто алгоритмических операций, представимых в виде некоторой формальной математической системы. Мы вернемся к этому вопросу в главе {10}.

Читатель может заметить определенное сходство между рассуждениями, устанавливающими, вопреки «недоказуемости», истинность $P_k(k)$, и парадоксом Рассела.[224] Помимо этого, наблюдается сходство и с доказательством Тьюринга о невозможности существования «машины Тьюринга»[225], которая могла бы решить проблему остановки. Эти сходства не случайны. Между этими тремя событиями имеется прочная историческая нить. Тьюринг пришел к своему доказательству после изучения работ Гёделя. Сам Гёдель был очень близко знаком с парадоксом

описывается с помощью вычислимых порядковых чисел. Есть несколько довольно естественных и легко формулируемых математических теорем, которые, если их пытаться доказать путем использования стандартных (введенных Пеано) правил арифметики, привели бы к «гипертрофированной» гёделевской процедуре (по числу шагов многократно превосходящей ту, что я описал ранее). Математические доказательства этих теорем по природе своей не зависят от туманных и сомнительных рассуждений, выходящих за рамки аппарата нормального математического доказательства (см. Сморински [1983]).

[223] **В.Э.:** Разумеется, не опирается. Кому вообще в голову может придти такая мысль, что эта «формалистская шелуха» играет какую-то роль в реальном мышлении?! Все математические истины вытекают из бокоанализа, осуществляемого одними (мозговыми) программами над другими программами.

[224] **В.Э.:** Нет, никакого такого сходства я не вижу. Парадокс Рассела имеет сходство с Парадоксом лжеца, потому что у обоих суть одна: зацикливание мозговой программы. А при работе с $P_k(k)$ никакого зацикливания нет.

[225] **В.Э.:** Да, такое сходство есть – и именно это и подрывает доверие к тому, что доказательство теоремы Гёделя может быть состоятельным.

Рассела и смог преобразовать те парадоксальные рассуждения, которые уводили слишком далеко в область логических абстракций, в состоятельное математическое доказательство. (Все эти утверждения уходят корнями к диагональному процессу Кантора, описанному в предыдущей главе, с. 80.)

Почему мы должны принимать доказательства Гёделя и Тьюринга и в то же время сбрасывать со счетов рассуждения, ведущие к парадоксу Рассела? Первые являются более ясными и безупречными с точки зрения математики, тогда как парадокс Рассела строится на более туманных рассуждениях об «огромных» множествах. Но нужно признать, что различия здесь не настолько очевидны, как нам хотелось бы. Попытка придать этим различиям ясность была лейтмотивом всей идеи формализма. Доказательство Гёделя, с одной стороны, показывает, что строгий формальный подход не выдерживает критики, но с другой стороны, оно не приводит нас к абсолютно надежной альтернативе.[226] По-моему, этот вопрос до сих пор не разрешен. Процедура, используемая в современной математике с целью избежать рассуждений, вовлекающих в рассмотрение «огромные» множества и приводящих к парадоксу Рассела, не является полностью удовлетворительной.[227] Более того, она, как правило, формулируется в чисто формалистских терминах – или же в терминах, которые не дают нам полной уверенности, что в результате их использования не возникнет противоречий.

Как бы там ни было, мне кажется, что из доказательства Гёделя следует с очевидностью, что понятие математической истины не может быть заключено ни в одну из формальных систем. Математическая истина выходит за рамки любого формализма. Возможно, это ясно даже без теоремы Гёделя. Иначе как бы мы решали, какие аксиомы и правила вывода брать в расчет при построении формальной системы? Нашим руководством в принятии такого решения должно всегда служить интуитивное понимание о том, что является «самоочевидно верным» с учетом «смысловых значений» символов системы. Как нам решить, какие формальные системы стоит использовать (в соответствии с нашим интуитивным ощущением «самоочевидности» и «смысла»), а какие – нет? Понятие «внутренней непротиворечивости» явно не подходит для этой цели. Можно иметь много внутренне непротиворечивых систем, которые «бессмысленны» с точки зрения их практического использования, в которых аксиомы и правила вывода имеют ложные в нашем понимании значения или же не имеют никаких. «Самоочевидность» и «смысл» – это понятия, которые потребовались бы даже без теоремы Гёделя.

Однако, без этой теоремы могло бы сложиться впечатление, что интуитивные понятия «самоочевидность» и «смысл» могли бы быть использованы только в самом начале раз и навсегда, просто чтобы изначально задать формальную систему, а затем мы могли бы отказаться от них при построении строгого математического доказательства для определения истины. Тогда, в соответствии с формалистскими воззрениями, эти «расплывчатые» интуитивные понятия задействовались бы только в «предварительных» размышлениях математиков, направленных на отыскание подходящего формального доказательства; а потом, когда дело дойдет до определения математической истины, они уже не играли бы никакой роли. Теорема Гёделя демонстрирует, что такой подход в действительности не является логически состоятельным в рамках фундаментальной философии математики. Понятие математической истины выходит за пределы всей теории формализма. В этом понятии есть нечто абсолютное и «данное свыше»[228]. И это как раз то, о чем трактует математический платонизм, обсуждаемый в конце предыдущей главы. Всякая формальная система имеет свойство сиюминутности и «человеко-зависимости». Такие системы, безусловно, играют очень важную роль в математических рассуждениях, но они могут указывать

---

[226] **В.Э.:** Если отказаться от (фундаментально ошибочного) пути аксиоматизации математики и перейти к тому, что предлагает Веданская теория (т.е. к обоснованию математики мозговыми программами), то положение в математике станет таким же, каково оно в науке программирования. А там нет этого вечного страха противоречий, там положение обыденно и спокойно. Рассуждайте об объектах математики так, как мы, программисты, рассуждаем о продуктах своих программ – и всё будет в порядке!

[227] Делается различие между «множествами» и «классами», где «множества» могут быть собраны вместе для образования других множеств или классов; а классы не могут образовывать сколько-нибудь более крупные объединения, будучи для этого «слишком большими». Однако не существует правила, согласно которому можно было бы решать, какие объединения могут рассматриваться как множества, а какие с необходимостью должны быть только классами – если не считать «порочно» замкнутое правило, гласящее, что множествами являются те объединения, которые можно составлять вместе, чтобы получать новые объединения!

[228] **В.Э.:** А точнее – данное мозговыми программами.

только частично верное (или приблизительное) направление к истине. Настоящая математическая истина выходит за пределы сотворенного человеком.

§4.5. Платонизм или интуиционизм?

Я указал две противостоящие друг другу школы математической философии, решительно причисляя себя более к платонистскому, нежели к формалистскому воззрению. В действительности же я применил довольно упрощенный подход при их разделении. Существует множество тонкостей, которые можно было бы принять в расчет. Например, в рамках платонизма можно поставить вопрос о том, существуют ли в реальности объекты математической мысли или это только лишь понятие «математической истины», которое является абсолютным. Я решил не обсуждать здесь подобные различия. В моем представлении абсолютность математической истины и платонистское существование математических понятий, по существу, тождественны. «Существование», которое должно быть приписано множеству Мандельброта, к примеру, есть свойство его абсолютной природы. Принадлежит ли точка плоскости Аргана множеству Мандельброта или нет – вопрос абсолютный, не зависящий от математика или компьютера, которые его исследуют. Эта «независимость-от-математика» множества Мандельброта и обеспечивает ему платонистское существование. Более того, наиболее тонкие детали этого множества лежат за пределами того, что можно достигнуть с помощью компьютера.[229] Эти устройства способны только аппроксимировать структуры, имеющие свое, более глубокое и «не зависящее-от-компьютера», существование. Я, однако, готов согласиться с тем, что имеются и прочие разумные точки зрения, с которых можно исследовать этот вопрос. Но здесь нам нет необходимости придавать значение этим различиям.

Есть также отличие в том, насколько далеко в своем платонизме готов зайти человек, провозглашающий свою принадлежность к этой школе. Сам Гёдель был глубоко убежденным платонистом. Математические выражения, которые я до сих пор рассматривал, являют собой довольно «мягкие» примеры того, что может встретиться в этом направлении.[230] Вполне возможны и более «запутанные» выражения, особенно в теории множеств. Когда рассматриваются все мыслимые ответвления этой теории, то порой возникают множества столь громадные и причудливо сконструированные,[231] что даже такой весьма убежденный платонист, как я, может начать сомневаться в абсолютности их существования (или, напротив, несуществования)[232]. Может наступить момент, когда определения множеств становятся настолько сложными и концептуально шаткими, что вопрос об истинности или ложности относящихся к ним математических выражений становится скорее субъективным и зависящим от мнения исследователя, нежели «ниспосланным свыше». Готов ли иной математик безоглядно следовать вместе с

---

[229] **В.Э.:** Говорил, говорил правильно, и – бац! – сбился. (Именно компьютеры – т.е. программы и алгоритмы – и создают множество Мандельброта).

[230] Континуум-гипотеза, которая упоминалась в главе 3, с. 81 (и из которой следует, что $C = \aleph_1$), является наиболее «экстремальным» математическим утверждением, которое здесь встречается (хотя часто рассматриваются и куда более «экстремальные» рассуждения). Континуум-гипотеза интересна еще и потому, что сам Гёдель, совместно с Полом Дж. Коэном, показал, что эта гипотеза в действительности не зависит от стандартных аксиом и правил теории множеств. Таким образом, отношение любого математика к континуум-гипотезе позволяет причислить его к сторонникам либо формалистской, либо платонистской точки зрения. Для формалиста данная гипотеза будет «недоказуемой», поскольку ее справедливость не может быть установлена или опровергнута, если опираться на стандартную (построенную Цермело и Френкелем) формальную систему, и, значит, не «имеет смысла» называть ее ни «истинной», ни «ложной». Однако, для убежденного платониста эта гипотеза является либо истинной, либо ложной, хотя какой именно – это можно установить только путем рассуждений некоторого нового типа, идущих еще дальше, чем использование гёделевских утверждений для формальной системы Цермело–Френкеля. (Коэн [1966] сам предложил принцип рефлексии, который позволяет показать, что континуум-гипотеза – «с очевидностью ложна»!)

[231] **В.Э.:** Много разных программ (рис. VE1) громоздят одну на другую. (Ничего особенного: любая компьютерная программная система, делающая сколь-нибудь приличную работу, состоит из тысяч подпрограмм; легко представить, во что превратилась бы работа программистов, если у них не было бы нормальных средств работы с программами – алгоритмических языков –, а приходилось бы – как бедным математикам – описывать всё это при помощи аксиом и правил вывода!)

[232] Живое и не слишком насыщенное техническими деталями изложение этой темы можно найти у Ракера [1984].

Гёделем путем платонизма, провозглашая истинность или ложность математических выражений, оперирующих подобными огромными множествами, всегда абсолютными (или «платонистскими») по своей природе; или же он, не заходя слишком далеко, будет говорить об абсолютности этих понятий лишь в том случае, если множества окажутся не слишком велики и довольно конструктивны. Ответ на этот вопрос не имеет большого отношения к нашей дискуссии. Множества (конечные или бесконечные), которые будут иметь для нас значение, по меркам вышеупомянутых множеств выглядят до смешного маленькими! Так что различия между разными платонистскими течениями нас волновать не должны.

Имеются, однако, и иные точки зрения в математике, такие как интуиционизм (и финитизм), которые, впадая в противоположную крайность, отказываются признавать существование каких бы то ни было бесконечных множеств.[233] Интуиционизм был основан в 1924 году датским математиком[234] Лейтзеном Э. Брауэром как альтернативный ответ – отличный от предлагаемого формализмом – на парадоксы (типа расселовского), которые могут возникать там, где бесконечные множества используются слишком вольно в математических рассуждениях. Зачатки этого подхода прослеживаются еще во времена Аристотеля, который, будучи учеником Платона, тем не менее отвергал его взгляды на абсолютное существование математических сущностей и возможность рассмотрения бесконечных множеств. Согласно интуиционизму, существование множества (бесконечного, равно как, впрочем, и конечного) не может признаваться как свойство, изначально ему присущее, а только лишь как функция правил, по которым оно организовано.

Характерная черта интуиционизма Брауэра состоит в отрицании закона «исключенного третьего». Этот закон говорит о том, что отрицание ложности некоторого выражения эквивалентно утверждению истинности этого выражения. (Или в принятой символике: $\sim(\sim P) \Leftrightarrow P$, отношение, которое нам уже встречалось ранее.) Наверное, Аристотель был бы очень недоволен, столкнувшись с отрицанием настолько логически «очевидного» факта! С общепринятых позиций здравого смысла закон «исключенного третьего» может рассматриваться как самоочевидная истина: если утверждение о том, что нечто ложно, само неверно, то это нечто должно быть непременно справедливым! (На этом законе основана математическая процедура «доказательства от противного», упомянутая на с. 63 {с.77 сноска 135 здесь}). Но интуиционисты считают допустимым отвергать справедливость этого закона. Основная причина здесь в том, что они занимают иную позицию по отношению к понятию существования, требуя, чтобы перед признанием существования математического объекта предъявлялось его конкретное (мысленное) построение. То есть, для интуиционалиста «существование» означает «конструктивное существование». В математическом доказательстве, использующем принцип «доказательства от противного», сперва выдвигается некая гипотеза, ложность которой затем устанавливается путем обнаружения противоречий, к которым приводят следствия из этой гипотезы. Эта гипотеза может принимать форму утверждения о том, что математический объект с требуемыми свойствами не существует. Когда это приводит к противоречию, то в обычной математике делается вывод о том, что данный объект да, существует. Но подобное доказательство, само по себе, не содержит руководства для построения такого объекта. Такое существование для интуициониста существованием отнюдь не является; и именно на этом основании они отказываются признавать закон «исключенного третьего» и процедуру «доказательства от противного». Сам Брауэр был совершенно неудовлетворен таким неконструктивным подходом к понятию существования.[235] Без указания реально осуществимого метода построения, говорил он,

---

[233] Интуиционизм был назван так потому, что ему предназначалось служить отражением человеческой мысли.

[234] **В.Э.:** Брауэр был голландским математиком (по-английски: *Dutch mathematician*).

[235] Сам Брауэр начал размышлять в этом направлении, в частности, потому, что очень придирчиво и болезненно относился к «неконструктивности» доказательства своей теоремы из области топологии, «теоремы Брауэра о неподвижной точке». Эта теорема утверждает, что, если вы возьмете круг – то есть окружность вместе со всеми точками внутри нее – и будете непрерывно двигать его внутри области, где он находился изначально, то найдется по крайней мере одна точка круга, – называемая неподвижной точкой, – которая окажется точно там же, откуда она начала движение. Не ясно, где именно располагается эта точка, и может ли их быть несколько – теорема говорит только о существовании такой точки. (Среди математических теорем существования, эта, на самом деле, носит довольно «конструктивный» характер. Примеры теорем существования более высокой степени неконструктивности – это теоремы, зависящие от так называемой «аксиомы выбора» или «леммы Цорна» (см. Коэн [1966], Ракер [1984]).) Трудность в случае

такая теория существования будет бессмысленной. В логике Брауэра нельзя сделать заключение о существовании объекта, исходя из ложности утверждения о его несуществовании!

По моему мнению, несмотря на похвальное стремление искать «конструктивное» решение вопроса о математическом существовании, интуиционизм, исповедуемый Брауэром, всё же является слишком радикальным. Брауэр впервые опубликовал свои идеи в 1924 году, более чем за десять лет до работ Тьюринга и Черча. Теперь, когда понятие конструктивности – в терминах теории Тьюринга о вычислимости – может изучаться в общепринятых рамках математической философии, уже нет необходимости впадать в крайности, как к тому нас призывает Брауэр. Мы можем исследовать конструктивность как самостоятельный предмет, отдельный от вопроса математического существования. Если мы последуем путем интуиционизма, то будем вынуждены отказаться от использования очень мощных приемов доказательства в математике, заметно ограничивая и лишая силы сам предмет.

Я не хочу излишне подробно останавливаться на разнообразных трудностях и кажущихся абсурдностях, к которым приводит интуиционистский подход; но упоминание некоторых проблем может оказаться полезным. Один из примеров, к которому часто обращается для иллюстрации Брауэр, касается дробной части числа π:

$$3{,}14152653589793\ldots\ .$$

Существует ли двадцать последовательных семерок где-нибудь в этой части, т.е.:

$$\pi = 3{,}14152653\ldots\ 77777777777777777777\ldots$$ [236]

или же нет? В обычных математических терминах всё, что мы можем сказать на сегодняшний день, это то, что они либо существуют, либо нет – и мы не знаем, какая из этих возможностей верна! Казалось бы, вполне безобидное утверждение. Однако правомерность утверждения «последовательность из двадцати семерок либо существует где-то в дробной части числа π, либо нет» будет отвергаться интуиционистами до тех пор, пока не получится установить (некоторым приемлемым с точки зрения интуиционизма конструктивным образом), что такая последовательность действительно существует, или же что такой последовательности нет! Прямого подсчета было бы достаточно для того, чтобы доказать, что данная последовательность действительно существует в дробной части π, но для доказательства невозможности ее существования потребовалась бы математическая теорема. Пока ни один компьютер не в состоянии просчитать дробную часть π с такой точностью, чтобы определить наличие там искомой последовательности. Можно было бы, с вероятностной точки зрения, предположить ее существование, однако, даже если бы компьютер вычислял каждую секунду, скажем, по 10^{10} цифр, то для нахождения этой последовательности потребовалось бы предположительно от ста до тысячи лет. Мне представляется гораздо более вероятным, что существование такой последовательности будет однажды установлено скорее математически, чем путем прямых вычислений (возможно, как побочный результат более глобального и интересного исследования) – хотя не исключено, что это будет сделано неприемлемым для интуицонистов способом!

Брауэра была аналогична той, что возникает в следующей задаче: найти точки, в которых f обращается в нуль, если известно, что f – действительная непрерывная функция действительной переменной, которая принимает как положительные, так и отрицательные значения. Стандартная процедура заключается в последовательном делении пополам отрезка, на котором функция меняет свой знак; но решение о том, какое именно промежуточное значение принимает функция (положительное, отрицательное или нулевое), может оказаться неконструктивным в том смысле, которого требует Брауэр.

[236] **В.Э.:** В (моем) английском оригинале: «*Does there exist a succession of ten consecutive sevens somewhere in this expansion, i.e.* π = *3.141592653589793.....7777777777....., or does there not*?» Откуда в русском тексте взялись <u>двадцать</u> семерок?! Или сам Пенроуз поменял в более позднем издании? Но ведь пример не Пенроуза, а Брауэра, но Брауэр давно умер (2 декабря 1966 года) и не мог изменить свой пример! (Число π тоже в русской книге неверно в пятом знаке после запятой... Причем два раза... (А в других местах правильно: см. §2.5 в первом томе и §3.2 в этом)). **P.S. (Через два часа)**. Похоже, Пенроуз все-таки исказил пример Брауэра, потому что его оценка вероятности встречи ряда семерок в десятичном представлении числа π в моем оригинале (копия главы из книги Академической библиотеки Латвии) звучит так: «*One's expectation on probabilistic grounds would be that such a succession does actually exist, but that even if a computer were to produce digits consistently at the rate of, say, one per second, it would be likely to take something of the order of between one hundred and one thousand years to find the sequence*!» Как видим, здесь он говорит о генерации одного знака в секунду, а в более позднем издании – уже о 10^{10} знаках в секунду; чтобы при этом те 100–1000 лет сохранились в силе, Пенроуз поменял 10 семерок на 20 семерок. Но у Брауэра было 10 семерок! Пример ведь известный – я его помню еще по книгам, которые читал 30 лет назад. Потому и стал исследовать, откуда же здесь взялись 20 семерок...

Данная проблема не имеет для математики особого значения и приведена лишь как наглядный пример. Брауэр, с позиций радикального интуиционизма, сказал бы, что в настоящее время утверждение «где-то в дробной части числа π существует двадцать последовательных семерок» не является ни справедливым, ни ложным. Если когда-либо в дальнейшем будет установлен конкретный результат – посредством вычислений или путем (интуиционистского) математического доказательства – то тогда утверждение станет «истинным» или «ложным», соответственно.[237] Сходный пример представляет собой и «последняя теорема Ферма». Вновь, согласно крайнему интуиционизму Брауэра, это утверждение не может быть сегодня признано ни ложным, ни истинным, но возможно, что его значение будет определено в будущем. По-моему, такая субъективность и «конъюнктурность» понятия математической истины просто неприемлема. Действительно, вопрос, будет ли – а если будет, то когда – официально признана «доказанность» некоторого математического результата, является весьма субъективным. Математическая истина не должна подчиняться такому «общественно-зависимому» критерию. Помимо этого, опираться на понятие математической истины, зависящее от времени – это, мягко говоря, наиболее неудобный и неудовлетворительный подход для математики, которую предполагается использовать для достоверного описания физического мира. Не все интуиционисты придерживаются таких радикальных взглядов, как Брауэр. И всё же точка зрения интуиционистов является, бесспорно, крайне неудобной, даже когда она родственна идеям конструктивизма. Немногие современные математики строго исповедуют чистый интуиционизм, даже если бы единственной причиной этого была бы его ограниченность относительно типов математических рассуждений, которые он позволяет использовать.

Я коротко описал три основных направления в современной математической философии: формализм, платонизм и интуиционизм.[238] Я не скрываю, что практически целиком отдаю предпочтение платонистской точке зрения, согласно которой математическая истина абсолютна и вечна, является внешней по отношению к любой теории и не базируется ни на каком «рукотворном» критерии; а математические объекты обладают свойством собственного вечного существования, не зависящего ни от человеческого общества, ни от конкретного физического объекта. Я попытался привести аргументы в пользу этой точки зрения в этом и предыдущем разделах, а также в конце третьей главы. Я надеюсь, что читатель готов следовать за мной и далее в этих рассуждениях, которые будут очень важны для понимания многих положений в дальнейшем.

§4.6. Теоремы гёделевского типа как следствие результатов, полученных Тьюрингом

В моем изложении теоремы Гёделя я опустил многие детали и к тому же оставил в стороне то, что относилось к неразрешимости вопроса о непротиворечивости системы аксиом и было исторически наиболее важной частью его доказательства. Моя задача состояла не в том, чтобы акцентировать внимание на «проблеме доказуемости непротиворечивости аксиом», столь важной для Гильберта и его современников; я стремился показать, что специфическое утверждение Гёделя – которое нельзя ни подтвердить, ни опровергнуть исходя из аксиом и правил вывода рассматриваемой формальной системы – оказывается с очевидностью верным, если опираться в наших рассуждениях на интуитивное понимание смысла применяемых процедур.

Я уже упоминал, что Тьюринг разработал свое доказательство неразрешимости проблемы остановки после изучения работ Гёделя. Оба доказательства имеют много общего и, естественно, основные положения из результатов Гёделя могут быть непосредственно получены путем

---

[237] **В.Э.:** Вообще не суть важно, характеризовать ли ситуацию словами «*утверждение «где-то в дробной части числа π существует десять последовательных семерок» не является ни справедливым, ни ложным*» или словами «*это утверждение истинно или ложно, только мы не знаем, каково именно оно*». Гораздо важнее просто представлять себе точно и ясно саму ситуацию, а именно: что речь идет о потенциальном продукте алгоритма, что после того, как этот алгоритм задан, его свойства объективны и не зависят от воли человека, но в то же время это всего лишь <u>потенциальный</u> продукт, не существующий нигде как физическая реальность. Человек, который (подобно мне) всё это ясно осознает, не чувствует никакой необходимости выбирать между мнением Брауэра и мнением платонистов (они оба правы).

[238] **В.Э.:** Из этих трех направлений самым слабым, бесперспективным и безнадежным, конечно, является формализм. Это вообще не математика, а искусственное третичное построение рядом со зданием математики. А платонизм и интуиционизм фактически объединяются Веданской теорией, которая, раскрыв подлинную сущность математики, стирает между ними границу.

использования процедуры Тьюринга. Давайте посмотрим, как это происходит, и как при этом можно несколько иным образом взглянуть на то, что осталось за кулисами теоремы Гёделя.

Непременное свойство формальной математической системы заключается в существовании вычислимого способа решить, является ли некоторая строка символов доказательством соответствующего математического утверждения или нет. Весь смысл формализации понятия математического доказательства в конечном счете сводится к тому, чтобы не требовалось никакого дополнительного суждения о состоятельности того или иного типа рассуждений. Необходимо обеспечить возможность проверять полностью механическим и заранее определенным способом, что предполагаемое доказательство и в самом деле является таковым – то есть должен существовать алгоритм для проверки доказательств. С другой стороны, мы не требуем существования алгоритмической процедуры нахождения доказательств истинности (или ложности) предлагаемых математических утверждений.

Как оказывается, алгоритм отыскания доказательства внутри произвольной формальной системы присутствует всегда, если только система допускает какое-нибудь доказательство. Действительно, мы прежде всего должны предполагать, что наша система формулируется на некотором языке символов, который можно выразить в терминах некоторого конечного «алфавита» символов. Как и ранее, давайте упорядочим наши строки символов лексикографически, что, как мы помним, означает расставление в алфавитном порядке строк каждой определенной длины, где все строчки единичной длины идут первыми, за ними следуют (также упорядоченные) строки из двух символов, потом – из трех, и так далее (с. 96).

Тогда мы будем иметь корректно построенные и пронумерованные в соответствии с лексикографической схемой доказательства.[239] Располагая нашим списком доказательств, мы одновременно имеем и перечень всех теорем нашей формальной системы, поскольку теорема – это утверждение, которое стоит в последней строчке списка корректно построенных доказательств.[240] Подобный перечень полностью проверяется непосредственными вычислениями: ведь мы можем рассматривать все строки символов системы – независимо от того, имеют они смысл как доказательства или нет – и начать тестировать нашим алгоритмом первую строчку, чтобы понять, является ли она доказательством, и отбросить ее, если нет; затем мы подобным же образом тестируем вторую строчку и исключаем ее, если и она не является доказательством; потом следует третья строчка, четвертая и так далее. Посредством этого мы в конце концов достигнем строки, содержащей доказательство, если таковая имеется в нашем списке.

Таким образом, если бы Гильберту удалось отыскать свою математическую систему – систему аксиом и правил вывода, достаточно мощную, чтобы позволить решать, путем формального доказательства, вопрос о справедливости или ложности любого математического утверждения, корректно сформулированного в рамках системы, – то тогда существовал бы общий алгоритмический метод выяснения истинности любого такого рассуждения. Почему это так? Потому что, если мы при помощи процедуры, описанной выше, находим искомое утверждение как последнюю строчку некоторого доказательства, то это утверждение автоматически считается доказанным. Если же, напротив, мы находим последнюю строчку, содержащую отрицание нашего утверждения, то мы тем самым доказываем его ложность. Если бы схема Гильберта была полной, то либо одна, либо другая возможность обязательно имела бы место (и если бы система была непротиворечивой, то обе возможности никогда бы не могли быть реализованы одновременно). То есть наша механическая процедура всегда бы прерывалась на некотором шаге и мы бы имели универсальный алгоритм для доказательства истинности или ложности всех утверждений системы. Это находилось бы в противоречии с результатами Тьюринга, изложенными во второй главе, согласно которым не существует общего алгоритма для доказательства математических утверждений. И, как следствие, мы доказали теорему Гёделя о том, что ни одна система наподобие задуманной Гильбертом не может быть полной в обсуждаемом нами смысле.

---

[239] **В.Э.:** Пенроуз только что упорядочивал строки символов; тем самым у него оказались упорядоченными в этом же списке и все доказательства! Стало быть, они действительно рассматривают всё доказательство (т.е. цепочку «высказываний») просто как одну строку! Боже, – если это так, то уж тут с бесконечностями будет каша неимоверная!

[240] **В.Э.:** Теперь доказательство опять не одна строка, а целое множество строк, в котором имеется последняя строка – теорема! Да черт побери, – вы вообще в состоянии хоть как-то разобраться со своими понятиями и терминами, или нет? (И вот эта белиберда еще будет претендовать на то, что она – вершина логики!)

В действительности теорема Гёделя носит более частный характер, поскольку от формальной системы того типа, который рассматривал Гёдель, требовалась адекватность по отношению к арифметическим утверждениям, а не математическим утверждениям вообще. Можем ли мы устроить так, чтобы все необходимые операции машины Тьюринга выполнялись только при помощи арифметики? Или, иными словами, могут ли все вычислимые функции натуральных чисел (т.е. рекурсивные, или алгоритмические функции – результаты действия машины Тьюринга) быть выражены в терминах обычной арифметики? На самом деле это так, хотя и не совсем. Нам понадобится одна дополнительная операция, которую мы добавим в систему стандартных правил арифметики и логики (включая кванторы $\exists$ и $\forall$). Эта операция просто выбирает «наименьшее натуральное число такое, что $K(x)$ имеет значение “истина”», где К( ) – заданная арифметически вычислимая функция исчисления высказываний, для которой предполагается существование такого числа, т.е. что $\exists\ x\ [K(x)]$ является истинным. (Если бы такого числа не было, то наша операция повторялась бы «бесконечно»[241] стараясь обнаружить несуществующее x.) В любом случае, предшествующие рассуждения показывают, что, исходя из результатов Тьюринга, программа Гильберта по сведению целых разделов математики к вычислениям в рамках некоторой формальной системы – невыполнима.

Как оказывается, эта процедура не может с очевидностью установить, что мы имеем утверждение Гёделя (наподобие $P_k(k)$), которое верно, но внутри системы недоказуемо. Однако, если вспомнить доказательство, приведенное в главе 2 и показывающее, «как “перехитрить” алгоритм» (с. 66 {с.81 здесь}), то мы увидим, что можно сделать нечто похожее и в этом случае. В том доказательстве мы смогли выяснить, что для любого алгоритма, определяющего момент остановки машины Тьюринга, можно придумать такое действие машины, которое не прекращается, хотя алгоритм – в отличие от нас – «увидеть» это не способен. (Вспомните, что мы требовали от алгоритма корректно информировать нас о моменте, когда машина Тьюринга действительно остановится, хотя мы допускаем, что он может не оповестить нас, если машина на самом деле не прекратит свое действие, продолжая работать вечно.) Таким образом, как и в ситуации с теоремой Гёделя, у нас есть утверждение (безостановочное действие машины Тьюринга), истинность которого мы можем установить при помощи интуитивного понимания, хотя определенная алгоритмическая процедура нам такой возможности и не дает.

§4.7. Рекурсивно нумеруемые множества

Существует способ для описания основных результатов, полученных Гёделем и Тьюрингом, в графическом виде, на языке теории множеств. Это позволит нам избежать произвольности описания в терминах конкретного символизма или в рамках формальной системы и выделить наиболее существенное. Мы будем рассматривать только множества натуральных чисел (конечные или бесконечные), такие как {4, 5, 8}, {0, 57, 100003}, {6}, {0}, {1, 2, 3, 4, ..., 9999}, {1, 2, 3, 4, ...}, {0, 2, 4, 6, 8, ...} и т.п.; или даже всё множество N = {0, 1, 2, 3, 4, ...}, равно как и пустое множество $\varnothing$ = {}. Нас будут интересовать только вопросы вычислимости, скажем: «Какие множества натуральных чисел могут быть сгенерированы с помощью алгоритма, а какие – нет?»

Чтобы сформулировать такой вопрос, мы можем считать, что каждое отдельное число n обозначает определенную строчку символов некоторой формальной системы. Это будет n-я строка символов, скажем, Q_n, согласно заданному в системе лексикографическому порядку («синтаксически корректных») утверждений. Тогда каждое натуральное число будет представлять некое утверждение. При этом множество всех утверждений формальной системы соответствует всему множеству натуральных чисел; а, допустим, теоремы этой системы будут составлять некоторое меньшее множество натуральных чисел, скажем, множество P. Однако детали произвольной системы нумерации утверждений для нас несущественны. Всё, что нам потребуется для установления соответствия между натуральными числами и утверждениями – это заданный алгоритм получения каждого утверждения Q_n (записанного должным образом в символических обозначениях) из отвечающего ему натурального числа n; и другой алгоритм для

[241] Вообще говоря, для нас является существенным, чтобы такие неудачные варианты могли реализоваться, тем самым гарантируя нам потенциальную возможность описывать любую алгоритмическую операцию. Вспомните, что для описания машин Тьюринга в общем мы должны допустить существование, в частности, машин, которые никогда не останавливаются.

получения n из Q_n. Имея эти алгоритмы в своем распоряжении, мы вольны идентифицировать множество натуральных чисел с множеством утверждений конкретной формальной системы.

Давайте выберем формальную систему достаточно непротиворечивую и широкую для того, чтобы включать в себя все действия всех машин Тьюринга – и, более того, «имеющую смысл» с учетом требования «самоочевидной справедливости» ее аксиом и правил вывода. Далее, пусть ряд утверждений Q_0, Q_1, Q_2, ... формальной системы имеет доказательства внутри системы. Эти «доказуемые» утверждения будут иметь номера, которые составляют некоторое множество в N – по сути, это множество P «теорем», рассмотренных выше. Мы уже видели, что существует алгоритм для последовательного построения всех утверждений произвольно заданной формальной системы, имеющих доказательства. (Как отмечено ранее, «n-е доказательство» Π_n получается из n алгоритмически. Всё, что нам надо – это посмотреть на последнюю строчку n-го доказательства, чтобы найти «n-е утверждение, доказуемое в рамках системы», т.е. n-ю «теорему».) Следовательно, мы имеем алгоритм последовательной генерации элементов P (при которой возможны и повторения, что для нас не важно).

Множество типа P, которое может быть построено с помощью некоторого алгоритма, называется рекурсивно нумеруемым. Заметьте, что множество утверждений, ложность которых может быть установлена в рамках системы – т.е. утверждений, чьи отрицания являются справедливыми – точно так же рекурсивно нумеруемо, поэтому мы можем просто нумеровать доказуемые утверждения по мере продвижения, учитывая и их отрицания. Есть большое число других, тоже рекурсивно нумеруемых, подмножеств N, для определения которых нам вовсе необязательно ссылаться на нашу формальную систему. Простыми примерами рекурсивно нумеруемых множеств могут служить множество четных чисел

$$\{0, 2, 4, 6, 8, \ldots\},$$

множество квадратов

$$\{0, 1, 4, 9, 16, \ldots\}$$

и множество простых чисел

$$\{2, 3, 5, 7, 11, \ldots\}.$$

Очевидно, мы можем построить любое из этих множеств при помощи алгоритма. Для каждого из этих трех примеров будет справедливо следующее свойство: дополнительное по отношению к рассматриваемому множество (состоящее из всех натуральных чисел, не входящих в исходное множество) является также рекурсивно нумеруемым. Дополнительными по отношению к вышеприведенным множествам будут, соответственно:

$$\{1, 3, 5, 7, 9, \ldots\},\ \{2, 3, 5, 6, 7, 8, 10, \ldots\} \text{ и } \{0, 1, 4, 6, 8, 9, 10, 12, \ldots\}.$$

Было бы достаточно просто указать алгоритм и для этих дополнительных множеств. Конечно же, мы можем выяснить алгоритмическим путем, является ли произвольное натуральное число n четным, квадратом натурального числа или простым числом, соответственно. Это дает нам алгоритм для построения обоих множеств, поскольку мы можем перебирать все натуральные числа и для каждого из них решать, принадлежит ли оно к определенному множеству или же к его дополнению. Множество, которое обладает свойством рекурсивной нумеруемости вместе со своим дополнением, называется рекурсивным. Очевидно, что дополнительное по отношению к рекурсивному множество также будет рекурсивным.

А существуют ли множества, которые рекурсивно нумеруемы, но рекурсивными, тем не менее, не являются? Давайте на минутку задумаемся над тем, какие следствия могут вытекать из подобного свойства. Поскольку элементы такого множества могут быть получены алгоритмическим путем, мы имели бы способ решить, принадлежит ли некоторый элемент – который, мы предполагаем, да, принадлежит множеству, – рассматриваемому множеству или нет. Всё, что от нас требуется, – это запустить алгоритм и прогонять его через все элементы множества до тех пор, пока он не найдет элемент, который мы ищем. Теперь давайте предположим, что искомый элемент не принадлежит данному множеству. В таком случае использование нашего алгоритма ничего не даст: он будет работать вечно, будучи не в состоянии прийти к решению. В этом случае нам потребуется алгоритм для построения дополнительного по отношению к исходному множества. Если этот алгоритм сможет обнаружить искомый элемент, то мы будем точно знать, что он не входит в состав исследуемого множества. Имея на вооружении оба алгоритма, мы так или иначе найдем данный элемент путем поочередного применения этих алгоритмов. Однако, такой благоприятный исход будет иметь место только в случае рекурсивного множества. Мы же предполагаем, что мы рассматриваем множество рекурсивно нумеруемое, но при этом не рекурсивное, т.е. наш предполагаемый алгоритм для построения дополнительного множества просто не существует! Таким образом, мы имеем курьезную ситуацию, когда можно определить, включен ли элемент в множество при условии, что он ему наверняка принадлежит; но в то же время нельзя гарантировать, что мы сможем это сделать посредством какого бы то ни было алгоритма для элементов, которые множеству не принадлежат.

Может ли возникнуть такая ситуация в реальности? Есть ли и вправду рекурсивно нумеруемые множества, не являющиеся рекурсивными? А как насчет множества *P*? Имеет ли это множество свойство рекурсивности? Мы знаем, что оно рекурсивно нумеруемо, так что нам остается только выяснить, будет ли также дополнительное к нему множество обладать этим свойством. Оказывается, что нет! Как мы можем сделать такой вывод? А давайте вспомним, что наряду с остальными операциями в нашей формальной системе разрешены и действия машин Тьюринга. Если мы обозначим *n*-ю машину Тьюринга через T_n, то выражение

«$T_n(n)$ останавливается»

– это утверждение – запишем его как $S(n)$, – которое мы можем сформулировать в рамках нашей системы для любого *n*. Утверждение $S(n)$ будет справедливым для одних значений *n*, и ложным – для остальных. Множество всех $S(n)$, образованное перебором натуральных чисел 0, 1, 2, 3, ..., будет представлено некоторым подмножеством N – скажем, *S*. Теперь учтем фундаментальный результат Тьюринга (глава 2, с. 63 {с.77 здесь}), который говорит о том, что не существует алгоритма, способного установить факт «$T_n(n)$ не останавливается» как раз в тех случаях, когда она действительно не останавливается. Это означает, что множество, состоящее из отрицаний $S(n)$, не является рекурсивно нумеруемым.

Мы видим, что часть *S*, принадлежащая *P*, состоит только из истинных $S(n)$. Почему это так? Понятно, что если любое конкретное $S(n)$ доказуемо, то оно должно быть верным (ведь наша формальная система была выбрана так, чтобы иметь «смысл»!), и поэтому часть *S*, лежащая в *P*, должна состоять исключительно из справедливых утверждений $S(n)$. Более того, ни одно верное утверждение $S(n)$ не должно лежать вне *P*, ибо, если $T_n(n)$ останавливается, то мы можем отыскать доказательство этому в рамках нашей системы.[242]

Теперь предположим, что дополнение *P* рекурсивно нумеруемо. Тогда у нас был бы алгоритм для построения элементов этого дополнительного множества. И мы смогли бы запустить его и пометить каждое утверждение $S(n)$, которое попадает в поле его действия. Это все будут ложные утверждения $S(n)$, так что наша процедура, по сути, обеспечит нам рекурсивную нумерацию множества таких утверждений. Но выше мы установили, что это множество не нумеруемо таким образом. Это противоречие показывает, что дополнение *P* все-таки не может быть рекурсивно пронумеровано; а *P*, следовательно, не является рекурсивным, что и требовалось доказать.

Эти свойства с очевидностью демонстрируют, что наша формальная система не может быть полной: то есть всегда будут существовать утверждения, чью справедливость (или ложность) невозможно доказать в рамках системы. Ведь если предположить, что такие «неразрешимые» утверждения не существуют, то дополнение множества *P* с необходимостью было бы множеством опровергаемых утверждений (всё, что недоказуемо, обязано быть опровергаемо). Но мы уже знаем, что опровергаемые утверждения составляют рекурсивно нумеруемое множество, что делает *P* рекурсивным. Однако, *P* не рекурсивно – противоречие, которое доказывает требуемую неполноту. Это основное утверждение теоремы Гёделя.

А как насчет подмножества *T* множества N, которое состоит из истинных утверждений нашей формальной системы? Рекурсивно ли *T*? Или оно только рекурсивно нумеруемо? А его дополнение? Оказывается, что ответ на все эти вопросы – отрицательный. Один из способов установить это – воспользоваться сделанным ранее выводом о невозможности алгоритмически сгенерировать ложные утверждения вида «$T_n(n)$ останавливается». Как следствие, ложные утверждения в целом не могут быть получены с помощью алгоритма, поскольку такой алгоритм, в частности, пронумеровал бы все вышеупомянутые ложные «$T_n(n)$ останавливается»-утверждения. Аналогично, и множество всех истинных утверждений не может быть построено при помощи алгоритма (так как любой подобный алгоритм легко модифицируется для нахождения ложных утверждений путем отрицания каждого из генерируемых им утверждений). Поскольку, тем самым, истинные утверждения не являются (равно как и ложные) рекурсивно

[242] Доказательство могло бы, в действительности, состоять из последовательности шагов, которые отражали бы действие машины, продолжающееся до ее остановки. Доказательство завершалось бы, как только машина остановится.

нумеруемыми, то они образуют гораздо более глубокий и сложноорганизованный массив, чем утверждения, имеющие доказательство внутри системы. И это иллюстрирует еще один аспект теоремы Гёделя: что понятие математической истины только частично досягаемо в рамках любой формальной системы.

Существуют некоторые простые классы истинных арифметических утверждений, которые всё же образуют рекурсивно нумеруемые множества. Например, как это нетрудно видеть, истинные утверждения вида

$$\exists\, w, x, ..., z\, [f\,(w, x, ..., z) = 0],$$

где f ( ) – некоторая функция, построенная из обычных арифметических операций сложения, вычитания, умножения и возведения в степень, составляют рекурсивно нумеруемые множества[243] (которые я обозначу через A). Пример утверждения такого рода – хотя мы не знаем, верно ли оно – это отрицание последней теоремы Ферма,[244] для которой мы можем взять за f ( ) функцию

$$f\,(w, x, y, z) = (x + 1)^{w+3} + (y + 1)^{w+3} - (z + 1)^{w+3}.$$

Однако, множество A не является рекурсивным (факт, который не так легко установить, хотя он и вытекает из оригинального доказательства Гёделя). Значит, мы не имеем никаких алгоритмических средств для выяснения – хотя бы в принципе – истинности или ложности последней теоремы Ферма.

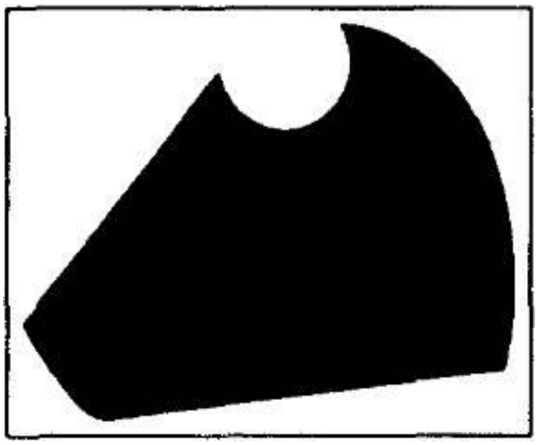


Рис. 4.1. Очень схематичное представление рекурсивного множества

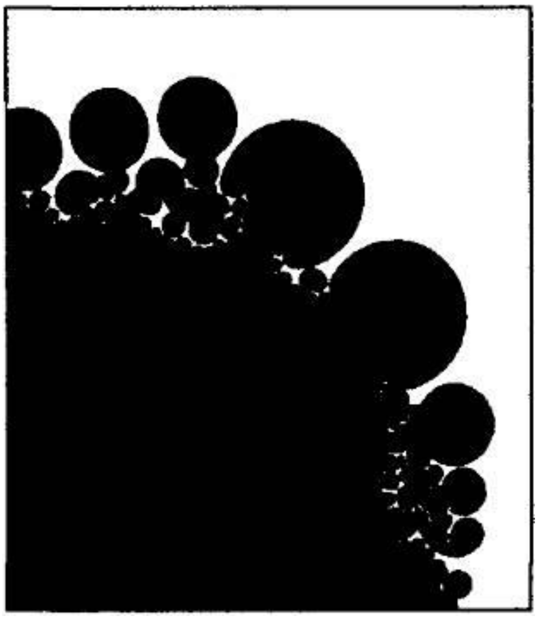


Рис. 4.2. Очень схематичное представление рекурсивно нумеруемого множества (темная область), которое не является рекурсивным. Здесь светлая область определяется только по «остаточному принципу», когда удаляется темная часть, построенная при помощи вычислений; а установить путем прямых вычислений, принадлежит ли заданная точка белой области, нельзя

На рис. 4.1 я попытался схематически представить рекурсивное множество как фигуру с простой и изящной границей, так что кажется, что определить непосредственно принадлежность произвольной точки этому множеству – дело несложное. Каждая точка на рисунке соответствует некоторому натуральному числу. При этом дополнительное множество также представлено в виде просто выглядящей области на плоскости.

На рис. 4.2 я постарался изобразить рекурсивно нумеруемое, но не рекурсивное множество в виде области со сложной границей, где подразумевается, что множество с одной стороны границы, – той, что рекурсивно нумеруема – должно выглядеть проще, чем с другой. Фигуры очень схематичны и не претендуют на какую бы то ни было «геометрическую аккуратность». И

[243] Мы нумеруем множества $\{v, w, x, ... , z\}$, где v представляет функцию f в согласии с некоторой лексикографической схемой. Мы (рекурсивно) проверяем на каждом этапе справедливость равенства

$$f\,(w, x, ..., z) = 0$$

и оставляем утверждение

$$\exists\, w, x, ..., z\, [(w, x, ..., z) = 0]$$

только в том случае, если это равенство выполняется.

[244] См. сноску 5 на с. 62. – *Прим. ред.*

конечно же, не стоит придавать большого значения тому, что эти рисунки изображены так, как если бы они были расположены на двумерной плоскости!

На рис. 4.3 я схематично обозначил, как расположены области *P*, *T* и *A* внутри множества N.

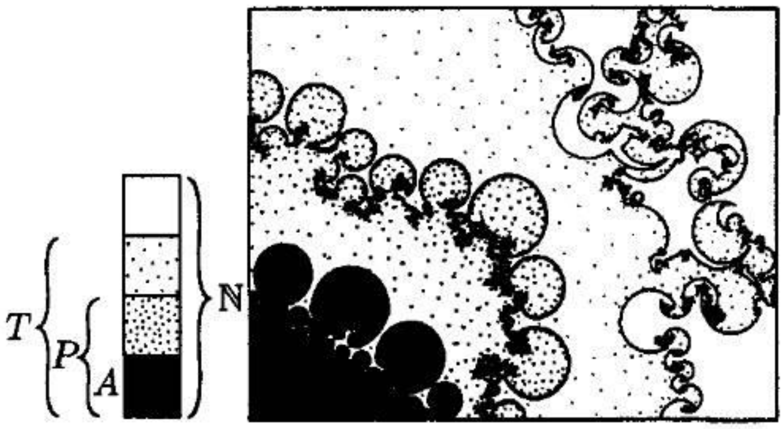


Рис. 4.3. Очень схематичное представление различных множеств утверждений. Множество *P* утверждений, доказуемых в рамках системы, является, как и *A*, рекурсивно нумеруемым, но не рекурсивным. Множество *T* истинных утверждений даже не рекурсивно нумеруемо

§4.8. Является ли множество Мандельброта рекурсивным?

Существенной характеристикой нерекурсивных множеств является их сложноорганизованность. Это свойство должно, в некотором смысле, препятствовать любым попыткам систематизации, которая, в противном случае, привела бы к некоторой «работающей» алгоритмической процедуре. Для нерекурсивного множества не существует общего алгоритмического пути к решению вопроса о принадлежности ему произвольного элемента (или «точки»). В начале третьей главы мы встретились с неким чрезвычайно сложно выглядящим множеством – с множеством Мандельброта. Хотя правила, по которым оно строится, поразительно просты, само множество представляет собой бесконечное разнообразие в высшей степени замысловатых структур. Может ли это быть примером настоящего нерекурсивного множества, явленного глазам смертных?

Читателю, однако, не понадобится много времени, чтобы сообразить, что эта парадигма сложности была создана специально для наших глаз волшебством вычислительных технологий с использованием современных быстродействующих компьютеров. А не являются ли компьютеры истинным воплощением алгоритмических действий? Конечно, это так, но всё же мы должны принимать во внимание способ, с помощью которого компьютеры, в действительности, создают эти картинки. Чтобы проверить, принадлежит точка плоскости Аргана – комплексное число c – множеству Мандельброта (закрашено черным) или его дополнению (светлая область), компьютер, начиная с нуля, применит отображение

$$z \to z^2 + c$$

сначала к $z = 0$, чтобы получить c; потом к $z = c$, чтобы получить $c^2 + c$; затем к $z = c^2 + c$, чтобы получить $c^4 + 2c^3 + c^2 + c$; и так далее. Если эта последовательность $0, c, c^2 + c, c^4 + 2c^3 + c^2 + c, ...$ остается ограниченной, то соответствующая точка c будет черной; в противном случае – белой. Как машина определяет, что такая последовательность остается ограниченной? В принципе, этот вопрос предполагает наличие информации о том, что происходит после <u>бесконечного</u> числа ее элементов! Сама по себе эта задача вычислительными методами не решается. К счастью, существуют способы предсказать исходя уже из конечного числа членов, когда последовательность станет неограниченной. (На самом деле, если последовательность достигает окружности радиуса $1 + \sqrt{2}$ с центром в начале координат, можно с уверенностью сказать, что она будет неограниченной.)

Таким образом, дополнение к множеству Мандельброта является, в некотором смысле, рекурсивно нумеруемым. Если комплексное число *c* расположено в светлой области, то существует алгоритм, подтверждающий этот факт. А как насчет самого множества Мандельброта – темного участка рисунка? Существует ли алгоритм, способный точно установить, что точка, принадлежащая предположительно темному участку, действительно ему принадлежит? Ответ на этот вопрос в настоящее время, похоже, отсутствует.[245] Я справлялся у многих коллег и экспертов, но ни один из них не слышал о подобном алгоритме. Равно как и никто из них не сталкивался с указанием на то, что такого алгоритма не существует. По крайней мере, насколько можно об этом судить, алгоритм для темной области на сегодняшний день неизвестен. Возможно, множество, дополнительное по отношению к множеству Мандельброта, действительно является примером рекурсивно нумеруемого, но не рекурсивного множества!

Прежде чем исследовать дальше это предположение, необходимо будет обсудить некоторые моменты, которые я ранее опускал. Эти вопросы будут довольно важны для нас в дальнейших рассуждениях по поводу вычислимости в физике. Я хотел бы заметить, что, на самом деле, я был несколько неточен в предшествующем изложении. Я применял такие понятия, как «рекурсивно нумеруемый» и «рекурсивный», к множествам точек в плоскости Аргана, т.е. множествам комплексных чисел. Но эти термины могут применяться только лишь для натуральных чисел и других счетных множеств. Мы видели в третьей главе (с. 80), что действительные числа не могут быть счетным множеством, равно как, следовательно, и комплексные – ведь любое действительное число может быть рассмотрено как частный случай некоторого комплексного числа с нулевой мнимой частью (с. 83). В действительности существует такое же «количество» комплексных чисел, как и действительных, а именно «С». (Чтобы установить взаимнооднозначное соответствие между комплексными и действительными числами, можно, грубо говоря, просто взять действительную и мнимую части комплексного числа (записанные в десятичной форме) и перемешать через одну поразрядно цифры из мнимой части с цифрами из вещественной, образуя, тем самым, действительное число: тогда, например, 3,6781... + *i*512,975... будет соответствовать действительному числу 50132,6977851....)

Дабы избежать этой проблемы, можно было бы ограничиться только вычислимыми комплексными числами, так как мы еще в третьей главе видели, что вычислимые действительные числа – а значит, и соответствующие им комплексные – являются счетными. Однако здесь кроется одна принципиальная трудность: не существует алгоритма, с помощью которого можно было бы сравнивать два вычислимых числа, полученных алгоритмически! (Мы можем алгоритмическим образом составить их разность, но мы не в состоянии будем выяснить, равна она нулю или нет. Представьте себе два алгоритма, которые генерируют цифры 0,99999... и 1,00000..., соответственно; мы никогда не узнаем, продолжаются ли нули и девятки в них до бесконечности – так, что числа оказываются равными – или же где-то в дробной части того или другого числа могут появиться иные цифры, делая эти числа неравными.) Таким образом, мы, возможно, никогда не сможем определить, равны ли между собой такие числа. Как следствие этого – наша неспособность решить даже в таком простом случае как единичный круг в плоскости Аргана (множество точек, лежащих на расстоянии не большем единицы от начала координат – черная фигура на рис. 4.4), лежит ли комплексное число в этом круге или нет. Трудность возникает не с точками, лежащими внутри или снаружи, а именно с точками на самой границе круга – то есть на самой единичной окружности. Эта окружность рассматривается по условию как часть круга. Предположим, что нам уже предоставлен в распоряжение алгоритм для получения цифр вещественной и мнимой

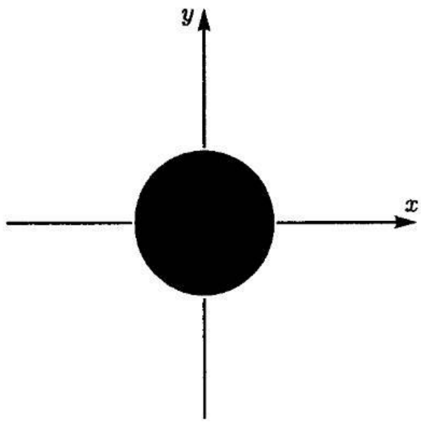


Рис. 4.4. Единичный круг, безусловно, должен рассматриваться как рекурсивное множество, но это требует определенных соглашений

[245] Недавно я узнал от Леоноры Блюм, что (заинтересовавшись моими комментариями в первом издании этой книги) она установила, что множество Мандельброта (и его дополнение) на самом деле являются, как я и предполагал, нерекурсивными в том смысле, который описан в десятом примечании. **В.Э.:** *У меня помечено ниже как *10.*

частей некоторого комплексного числа. Если мы предполагаем, что это комплексное число лежит на единичной окружности, то мы не можем с необходимостью подтвердить этот факт. Не существует алгоритма, чтобы установить, является ли вычислимое число

$$x^2 + y^2$$

равным единице, что служит критерием для принадлежности комплексного числа $x + iy$ данной единичной окружности.

Очевидно, это совсем не то, что нам нужно. Единичный круг, безусловно, должен рассматриваться как рекурсивное множество. Едва ли найдется сколь-нибудь значительное число множеств, более простых, чем единичный круг! Чтобы обойти эту проблему, одним из способов может быть игнорирование границы. Ведь для точек, лежащих внутри (или снаружи), безусловно существует алгоритм, устанавливающий этот факт. (Можно просто последовательно генерировать цифры числа $x^2 + y^2$, и, в конце концов, мы найдем цифру, отличную от 9 в дробной части 0,9999... или отличную от 0 – в дробной части 1,00000... .) В этом смысле единичный круг является рекурсивным. Но этот подход чрезвычайно неудобен для математики, поскольку там часто возникает необходимость ссылаться в рассуждениях на то, что происходит именно на границах. С другой стороны, вполне возможно, что такая точка зрения окажется применимой в области физики. Позднее нам еще придется вернуться к этому вопросу.

Существует другой метод, имеющий непосредственное отношение к данному вопросу, который не предполагает вообще обращения к вычислимым комплексным числам. Вместо того, чтобы пытаться пронумеровать комплексные числа внутри или снаружи рассматриваемого множества, мы просто будем вызывать алгоритм, который для любого наперед заданного комплексного числа будет определять, принадлежит оно нашему множеству или же его дополнению. Говоря «наперед заданный», я подразумеваю, что для каждого числа, которое мы рассматриваем, нам некоторым – быть может, «волшебным» – образом известны цифры мнимой и вещественной части, одна за другой, и в таком количестве, сколько нам нужно. Я не требую, чтобы существовал алгоритм, известный или неизвестный, для нахождения этих цифр. Множество комплексных чисел считалось бы «рекурсивно нумеруемым», если бы существовал хотя бы единственный алгоритм такой, что для любой заданной ему вышеуказанным образом последовательности цифр он бы говорил «да» после конечного числа шагов тогда и только тогда, когда комплексное число действительно принадлежит этому множеству. Оказывается, что как и в случае подхода, предложенного выше, эта точка зрения также «игнорирует» границы. Следовательно, внутренняя и внешняя области единичного диска будут каждая по отдельности считаться рекурсивно нумеруемыми в указанном смысле, тогда как сама граница – нет.

Для меня совершенно не очевидно, что какой-либо из этих методов дает то, что нам нужно.[246] Философия «игнорирования границ», будучи приложенной к множеству Мандельброта, может привести к потере большого числа тонких моментов. Одна часть этого множества состоит из «клякс» – внутренних областей, а другая – из «усиков». Наибольшие сложности при этом связаны, видимо, с «усиками», которые могут «извиваться» самым причудливым образом. Однако, «усики» не принадлежат внутренней части множества, и, тем самым, они были бы проигнорированы, используй мы любой из двух вышеприведенных подходов. Но даже при таком допущении остается неясность, можно ли считать множество Мандельброта рекурсивным в том случае, когда рассматриваются только «кляксы». Похоже, что вопрос этот связан с некоторым недоказанным предположением, касающимся самого множества, а именно: является ли оно, что называется, «локально связным»? Я не собираюсь здесь разбирать значение этого понятия или его важность для данного вопроса. Я хочу просто показать, что существует ряд трудностей, которые вызывают неразрешенные на сегодняшний день вопросы, касающиеся множества Мандельброта, чье решение – первоочередная задача для некоторых современных математических исследований.

Существуют также и другие подходы, которые могут использоваться с тем, чтобы обойти проблему несчетности комплексных чисел. Вместо того, чтобы рассматривать все вычислимые комплексные числа, можно ограничиться только подмножеством таких чисел, для любой пары которых можно вычислительным путем установить их равенство. Простым примером такого

---

[246] (*10) Блюмом (**В.Э.:** *может ошибка перевода и это Леонора Блюм*? *См.* {Лит.}), Шубом и Смэйлом [1989] была разработана новая теория вычислимости для действительных функций от действительных переменных (в отличие от общепринятых функций натуральных чисел, принимающих натуральные значения), подробности которой я узнал лишь совсем недавно. Эта теория применима и к комплексным функциям, а кроме того, может сыграть заметную роль в упомянутых мной вопросах.

подмножества могут служить «рациональные» комплексные числа, у которых как мнимая, так и вещественная части могут быть представлены рациональными числами. Я не думаю, однако, что это дало бы многое в случае «усиков» множества Мандельброта, поскольку такая точка зрения накладывает очень значительные ограничения. Более удовлетворительным могло бы оказаться рассмотрение алгебраических чисел – тех комплексных чисел, которые являются алгебраическими решениями уравнений с целыми коэффициентами. Например, все решения z уравнения

$$129z^7 - 33z^5 + 725z^4 + 16z^3 - 2z - 3 = 0$$

– это алгебраические числа. Такие числа будут счетными и вычислимыми, и задача проверки двух из них на равенство будет решаться путем прямого вычисления. (Как выясняется, многие из них будут лежать на границе единичного круга и «усиков» множества Мандельброта.) И мы можем по желанию рассматривать вопрос о рекурсивности множества Мандельброта в терминах этих чисел.

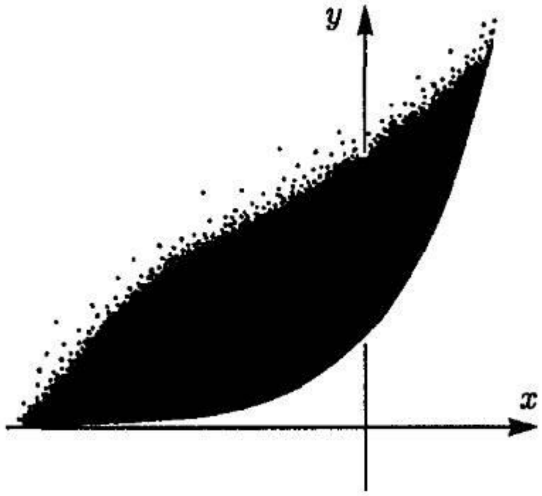


Рис. 4.5. Множество, определенное экспоненциальным соотношением $y \geq e^x$, должно также рассматриваться как рекурсивное

Возможно, что алгебраические числа оказались бы подходящим инструментом для двух обсуждаемых нами множеств, но они не снимают все наши трудности в общем случае. Пусть мы рассматриваем множество (темная область на рис. 4.5), определяемое неравенством

$$y \geq e^x,$$

где $x + \mathrm{i}y (= z)$ – точка в плоскости Аргана. Внутренняя часть множества, равно как и внутренняя часть его дополнения, будут рекурсивно нумеруемыми в соответствии с любой из вышеизложенных точек зрения, но (как следует из знаменитой теоремы Ф. Линдеманна, доказанной в 1882 году) граница, $y = e^x$, содержит только одну алгебраическую точку, а именно точку $z = \mathrm{i}$. В этом случае алгебраические числа никак не могут нам помочь при исследовании алгоритмической по своей природе границы! Несложно определить другой подкласс вычислимых чисел, которые будут подходить в данном конкретном случае, но при этом всё равно останется ощущение, что правильный подход нами до сих пор так и не был найден.

§4.9. Некоторые примеры нерекурсивной математики

Существует немало областей математики, где возникают проблемы нерекурсивного характера. Это означает, что мы можем сталкиваться с задачами, ответ к которым в каждом случае либо «да», либо «нет», но определить, какой из них верен, – нельзя из-за отсутствия соответствующего общего алгоритма. Некоторые из этих классов задач выглядят на удивление просто.

Например, рассмотрим задачу об отыскании целочисленных решений системы алгебраических уравнений с целыми коэффициентами. Эти уравнения известны под именем диофантовых (в честь греческого математика Диофанта, который жил в третьем веке до нашей эры и изучал уравнения такого типа). Подобные уравнения выглядят, например, как

$$z^3 - y - 1 = 0,$$
$$yz^2 - 2x - 2 = 0,$$
$$y^2 - 2xz + z + 1 = 0,$$

и задача состоит в том, чтобы определить, могут ли они быть решены в целых x, y, z. Оказывается, что в этом конкретном случае существует тройка целых чисел, дающая решение этой системы:

$$x = 13, y = 7, z = 2.$$

Но для произвольной системы диофантовых уравнений никакого алгоритма не существует.[247] Арифметика Диофанта, несмотря на простоту входящих в нее выражений, является частью неалгоритмической математики!

(Несколько менее тривиальным является пример топологической эквивалентности многообразий. Я упоминаю об этом только вкратце, ибо в главе {8} будут рассматриваться вопросы, имеющие к данному определенное отношение. Чтобы понять, что такое «многообразие», представьте для начала петлю, которая является многообразием в одном измерении; затем представьте замкнутую поверхность – многообразие в двух измерениях. Далее попробуйте представить некую «поверхность», имеющую три и более измерений. «Топологическая эквивалентность» двух многообразий означает, что одно из них может быть деформировано в другое путем непрерывных преобразований – без разрывов и склеек. Так, сфера и поверхность куба являются топологически эквивалентными, хотя они не эквивалентны поверхности кольца или чашки с ручкой – хотя последние топологически эквивалентны друг другу. При этом для двумерных многообразий существует алгоритм, позволяющий определить, эквивалентны ли произвольные два многообразия друг другу или нет – в сущности, заключающийся в подсчете «ручек», которые имеет каждая из поверхностей. Для случая трех измерений вопрос о существовании такого алгоритма на момент написания книги остается без ответа; однако для четырех и более измерений уже известно, что такого алгоритма быть не может. Возможно, четырехмерный случай имеет некое отношение к физике, поскольку согласно теории общей относительности Эйнштейна пространство и время совместно образуют четырехмерное многообразие (см. главу 5, с. {163}). Герох и Хартли в 1986 году высказали предположение о том, что свойство неалгоритмичности может иметь отношение к «квантовой гравитации» (см. также главу {8}).)

Давайте теперь рассмотрим иной тип задач, называемых задачами со словами.[248] Допустим, у нас есть некий алфавит символов, и мы рассматриваем различные строки этих символов, трактуя их как слова. Слова могут сами по себе не иметь никакого значения, но мы должны иметь некоторый (конечный) список «равенств» между ними, которые мы сможем использовать для дальнейшего построения таких «равенств». Это делается путем подстановки слов из исходного списка в другие (как правило, более длинные) слова, которые содержат их в виде составных частей. Каждая такая часть может быть заменена на равную ей в соответствии с используемым списком. Тогда для любой данной пары слов мы должны решить задачу об их равенстве согласно этим правилам.

В качестве примера мы можем взять для нашего исходного списка, скажем, такие равенства:

EAT = AT
ATE = A
LATER = LOW
PAN = PILLOW
CARP = ME.

Отсюда мы можем, например, вывести

LAP = LEAP,

используя последовательные замены из второго, первого и снова второго соотношения из нашего исходного листа:

LAP = LATEP = LEATEP = LEAP.

Проблема теперь заключается в том, чтобы выяснить, сможем ли мы для любой наперед заданной пары слов осуществить вышеописанным образом переход от одного из них к другому? Можем мы перейти от CATERPILLAR к MAN, или, скажем, от CARPET – к MEAT? Ответ в первом случае оказывается утвердительным, тогда как во втором – отрицательным. Когда ответ

---

[247] Это дает (отрицательный) ответ на десятую проблему Гильберта, упомянутую на с. {44} (см., например, Дэвлин [1988]). Здесь количество переменных неограниченно. Однако, известно, что для выполнения свойства неалгоритмичности достаточно и девяти.

[248] Эта конкретная задача называется (если быть более точным) «задачей со словами для полугрупп». Существуют также и другие разновидности этой задачи, в которых действуют несколько отличные правила. Но нас это сейчас волновать не должно.

утвердителен, стандартный путь показать его справедливость заключается в построении цепочки равенств, где каждое из слов получается из предыдущего с учетом допустимых соотношений. Итак, имеем (обозначая буквы, назначенные к замене, жирным шрифтом, а только что измененные – курсивом)[249]:

C**ATE**RPILLAR = C*A*RPILL**A**R = CARPIL**L*ATE*R** = CAR**PIL*LOW*** = **CAR***PAN* = *ME***A**N = M**E*ATE***N = M***AT*E**N = M*A*N.

Как мы можем утверждать, что посредством разрешенных подстановок невозможно получить MEAT из CARPET? Для демонстрации этого факта придется подумать чуть больше, однако показать это не так уж сложно, причем множеством разных способов. Простейшим представляется следующий: в каждом «равенстве» из нашего списка число букв A плюс число букв W плюс число букв M с каждой стороны одинаково. Значит, общая сумма указанных букв не может меняться в процессе преобразования по допустимым нашим списком правилам. Однако, для CARPET эта сумма равна 1, а для MEAT – 2. Следовательно, не существует способа получить из первого слова второе при помощи вышеприведенного списка равенств.[250]

Заметьте, что когда два слова «равны», мы можем показать это, просто приведя допустимую формальную строчку символов, построенную с помощью заданных нами правил; тогда как в случае их «неравенства» мы должны прибегать к рассуждениям об этих самых правилах. Существует четкий алгоритм, который мы можем использовать для установления «равенства» между двумя словами в том случае, когда они действительно «равны». Всё, что нам требуется, это составить лексикографический перечень всех возможных последовательностей слов, и потом вычеркнуть из этого списка любую строчку, где имеется пара слов, в которой последующее нельзя получить из предыдущего при помощи какого бы то ни было правила из исходного списка. Оставшиеся последовательности дадут нам набор всех искомых «равенств» между словами. Однако, в общем случае нет такого явного алгоритма для случая, когда два слова «неравны», и нам, возможно, пришлось бы применить «интеллект» для установления этого факта. (Конечно же, мне потребовалось некоторое время, прежде чем я заметил описанный выше «трюк»[251], при помощи которого доказал, что CARPET и MEAT «неравны». А для другого примера «трюк» мог бы понадобиться совершенно иной. Кстати, интеллект помогает – хотя и не обязательно – и в случае, когда необходимо установить существование некоторого «равенства».)

В действительности, для нашего конкретного списка из пяти «равенств», которые составляют исходный список в рассмотренном выше примере, привести алгоритм, устанавливающий «неравенство» в случае, когда два слова и вправду «неравны» – не так уж сложно. Однако, чтобы отыскать алгоритм в этом случае, потребовалась изрядная работа интеллекта! И, конечно, оказывается, что не существует единого универсального алгоритма для всех возможных вариантов исходного списка. Общая задача со словами принадлежит области нерекурсивной математики![252]

Существуют даже определенные варианты выбора исходного списка, для которых нет алгоритма решения задачи сравнения двух слов. Один из них дается таким набором:

AH = HA
OH = HO
AT = TA
OT = TO
TAI = IT
HOI = IH
THAT = ITHT.

(Этот список взят из списка, предложенного Григорием Цейтиным и Даной Скотт в 1955 году (см. Гарднер [1958]).) Таким образом, эта частная задача со словами служит примером нерекурсивной математики в том смысле, что, используя такой исходный список, мы не можем алгоритмическим путем решить, «равны» два наперед заданных слова или нет.

---

[249] **В.Э.:** Я болды усилил еще и подчеркиванием, а курсивы – красным цветом. В русской книге действительные болды и курсивы не соответствовали установкам Пенроуза и являли собой полную бессмыслицу. Я восстановил всё, как надо.

[250] **В.Э.:** Однако этот факт был установлен алгоритмом, примененным Пенроузом.

[251] **В.Э.:** Нашел алгоритм.

[252] **В.Э.:** Так что такое «нерекурсивная математика»? Просто набор задач, для которых нет общего алгоритма решения и приходится обходиться частными случаями?

Общая задача со словами возникает как следствие рассмотрения формализованной математической логики («формальных систем» и т.п., в соответствии с обсуждаемым ранее). Исходный список выполняет роль системы аксиом, а правила замены слов – правил вывода. Доказательство нерекурсивности задачи со словами вытекает из подобных рассуждений.

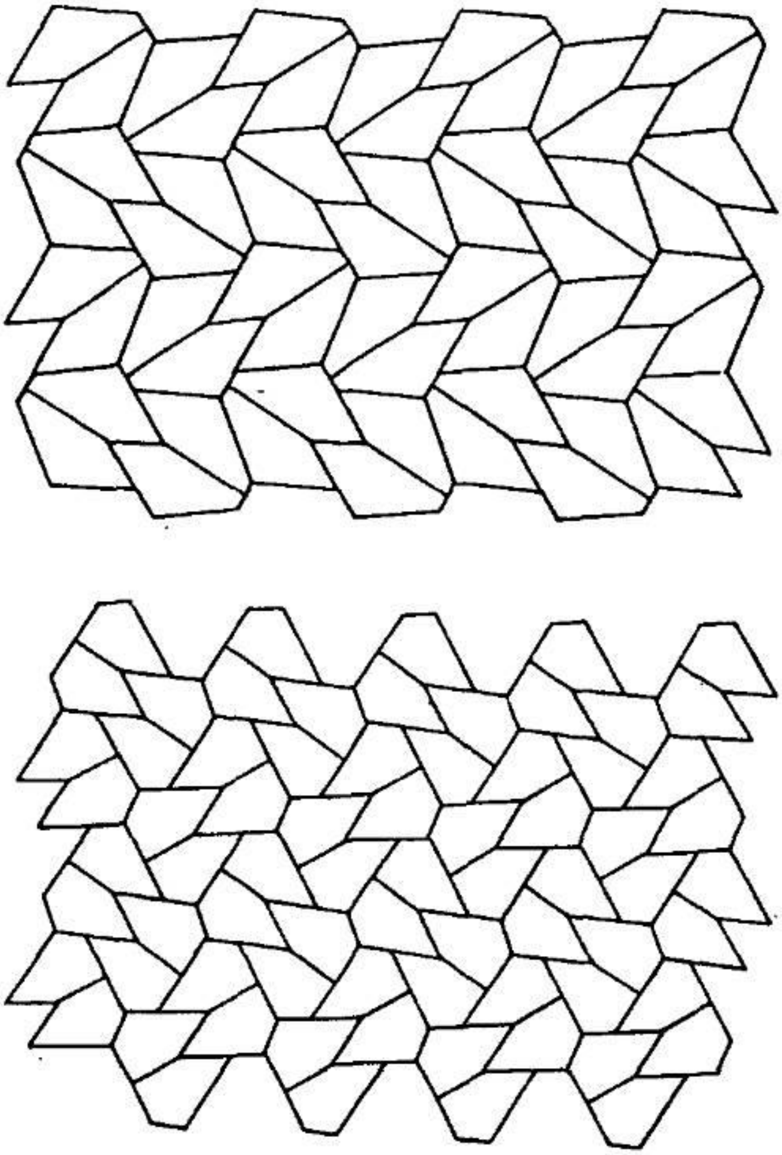


Рис. 4.6. Два примера периодического замощения плоскости фигурой одной формы (предложены Марджори Райс (Marjorie Rice) в 1976 году)

В качестве последнего примера задачи из области нерекурсивной математики давайте рассмотрим вопрос о покрытии Евклидовой плоскости многоугольниками, разнообразие форм которых ограничено, а сам вопрос при этом ставится так: можем ли мы выложить всю плоскость полностью, без разрывов и нахлестов, используя фигуры только данных нам форм? Такая укладка фигур называется замощением плоскости. Мы знаем, что такое замощение возможно при помощи только квадратов, только равнобедренных треугольников или только правильными шестиугольниками (как изображено на рис. {10.2} на с. 351), но невозможно, если использовать только правильные пятиугольники. Многими иными фигурами, такими, как два неправильных пятиугольника на рис. 4.6, также можно выложить плоскость. Замощение фигурами двух форм может стать более хитроумной задачей. Два простых примера даны на рис. 4.7. Все эти замощения являются периодическими; это означает, что они в точности повторяются по всей плоскости в двух независимых направлениях. На языке математики мы бы сказали, что существует параллелограмм периодов – параллелограмм, который, будучи неким образом выделен и затем повторен снова и снова в двух направлениях, параллельных его сторонам, даст в результате заданный узор покрытия. На рис. 4.8 представлен пример, где периодическое покрытие слева состоит из «плиток» в форме шипов, а справа указан соответствующий параллелограмм периодов.

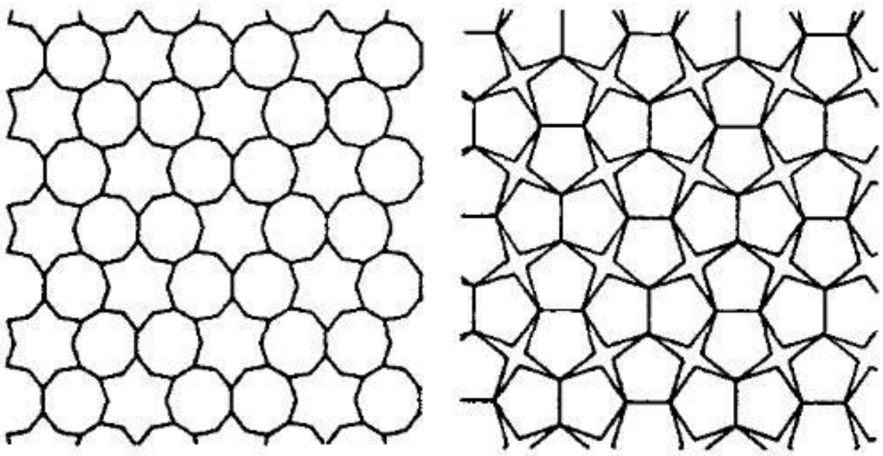


Рис. 4.7. Два примера периодического замощения плоскости фигурами двух форм

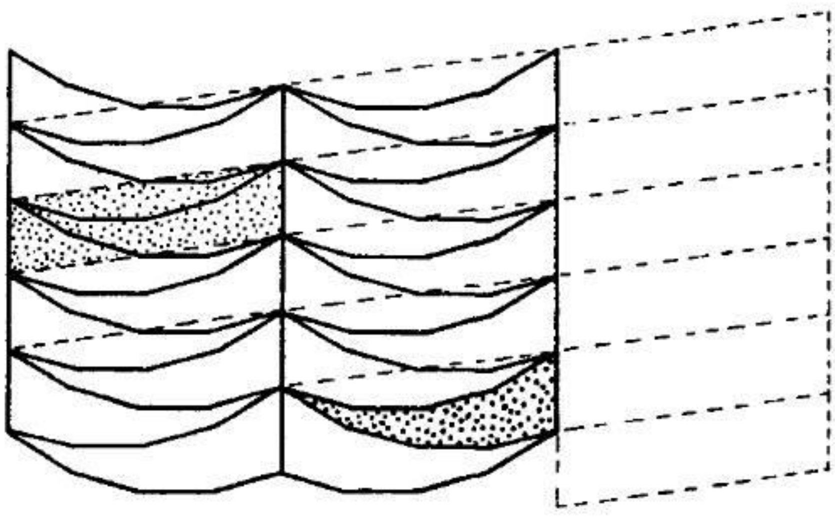


Рис. 4.8. Периодическое замощение и его параллелограмм периодов

С другой стороны, существует множество типов замощений плоскости, которые не являются периодическими. Рис. 4.9 изображает три непериодических «спиральных» замощения из таких же шиловидных «плиток», как и на рис. 4.8. Эта форма «плиток», известная как «универсальная» (по вполне понятным причинам!), была предложена Б. Грюнбаумом и Дж.К. Шепардом [1981, 1987] на основании форм, изученных Х. Фодербергом. Обратите внимание, что универсальная форма позволяет замостить плоскость как периодически, так и непериодически. Это свойство характерно и для многих других форм единичных «плиток» и наборов «плиток». А могут ли существовать «плитки» (или конечные наборы «плиток»), которые бы покрывали плоскость только непериодически? Ответ на этот вопрос будет «да». На рис. 4.10 я изобразил сконструированный американским математиком Рафаэлем Робинсоном набор из фигур шести различных форм, которым можно замостить всю плоскость, но только непериодическим образом.

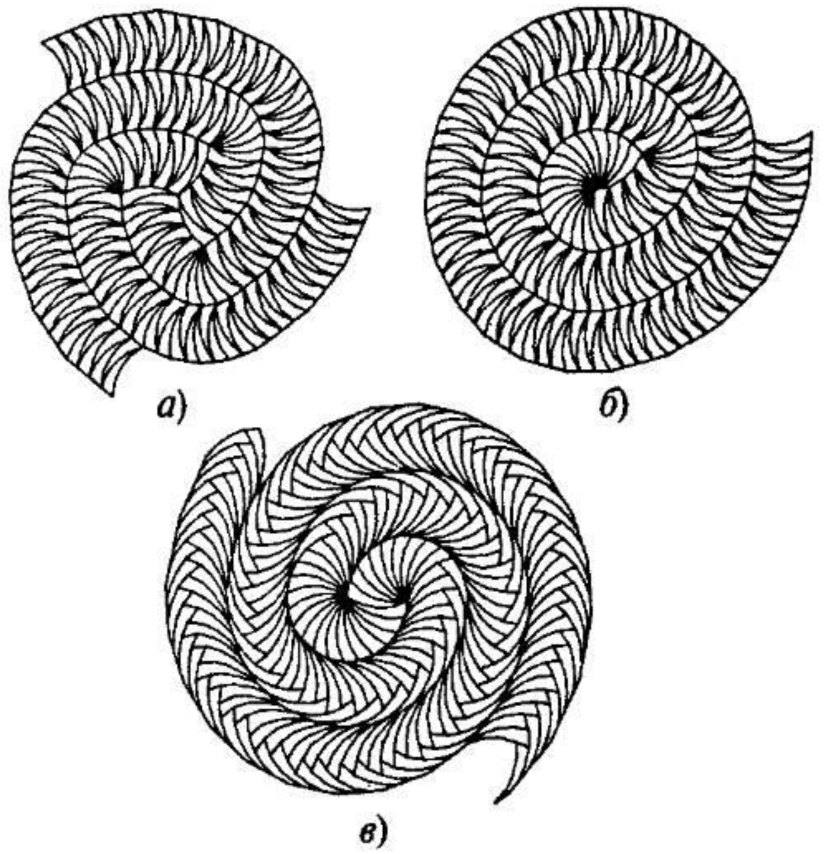


Рис. 4.9. Три непериодических «спиральных» замощения из таких же «универсальных» плиток, как и на рис.4.8

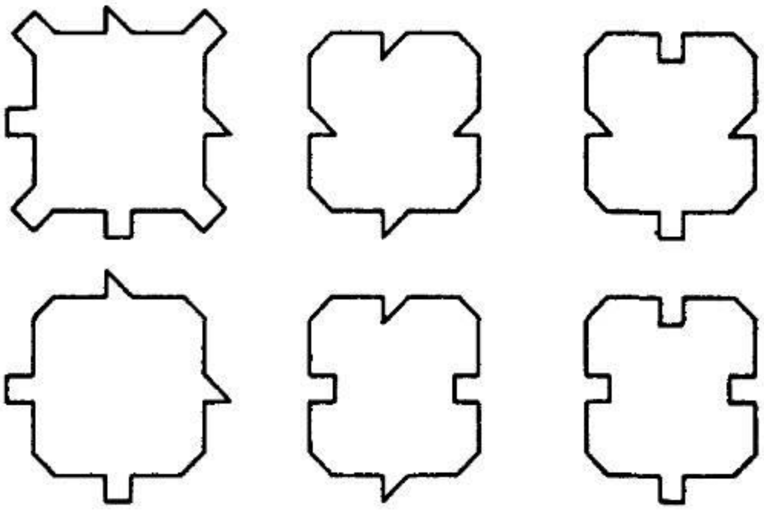


Рис. 4.10. Набор Рафаэля Робинсона из шести плиток, который покрывает плоскость только непериодически

Небесполезно было бы сделать историческое отступление и посмотреть, как появился этот непериодический набор (см. Грюнбаум, Шепард [1987]). В 1961 году американский логик китайского происхождения Хао Ванг поставил вопрос о существовании процедуры для решения задачи замощения, или, иными словами, о нахождении алгоритма, который позволил бы выяснить возможность замощения всей плоскости с помощью конечного набора многоугольников различной формы![253] Ему удалось показать, что такая процедура могла бы существовать, если бы получилось доказать следующую гипотезу: любой конечный набор разных «плиток», с помощью которого можно каким-нибудь способом выполнить замощение плоскости, пригоден также и для ее периодического замощения. Мне думается, в то время интуитивно казалось, что не может существовать набор «плиток», нарушающий это условие (т.е. не может существовать «непериодический» набор плиток). Однако в 1966 году, следуя в указанном Хао Вангом направлении, Роберт Бергер смог показать, что, на самом деле, процедуры решения задачи покрытия не существует: эта задача также принадлежит области нерекурсивной математики![254]

С учетом доказанного Хао Вангом это означало, что хотя бы один непериодический набор «плиток» должен существовать; и Бергер смог построить первый такой набор. Однако, из-за сложности выбранного им способа рассуждений, его набор состоял из ненормально большого числа «плиток» разной формы – изначально их насчитывалось 20˙426. Использовав некоторый дополнительный искусный прием, Бергеру удалось сократить это число до 104. А в 1971 году Рафаэль Робинсон довел его до шести, которые изображены на рис. 4.10.

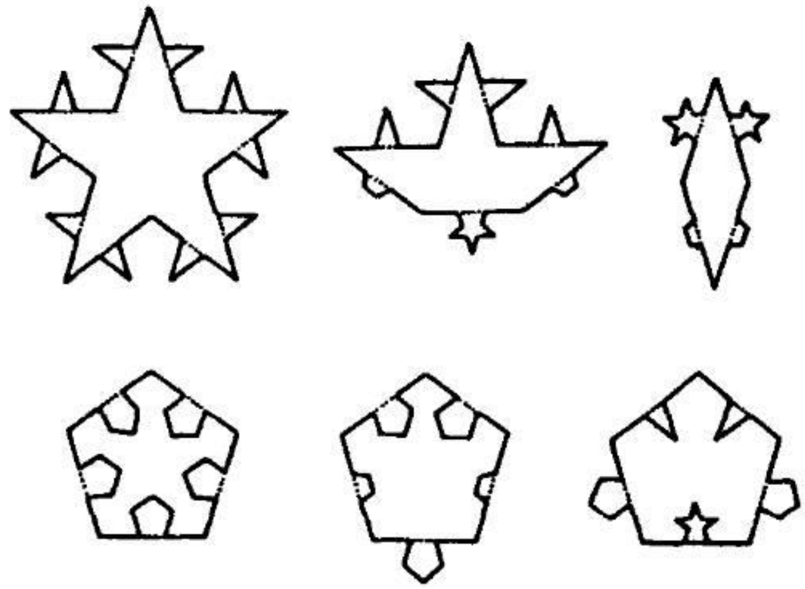


Рис. 4.11. Другой набор из шести плиток, который покрывает плоскость только непериодически

Другой непериодический набор из шести «плиток» представлен на рис. 4.11. Это множество я придумал сам в 1973 году, следуя в своих рассуждениях несколько отличным путем. (Я вернусь к этой теме в главе 10, где на рис. {10.3}, с. 351, изображен массив, покрытый такими «плитками».) После того, как я познакомился с «шестиплиточным» набором Робинсона, я начал думать о том, как сократить их число; и путем различных манипуляций с разрезаниями и склеиванием я, в конечном счете, смог довести количество «плиток» до двух. Две альтернативные схемы представлены на рис. 4.12. Узоры, которые получаются в результате полного замощения и имеющие с необходимостью непериодическую структуру, обладают рядом замечательных свойств, в том числе – кажущейся невозможной с точки зрения кристаллографии квазипериодической симметрией с осью пятого порядка. К этому вопросу я вернусь позднее.

Вероятно, это покажется удивительным, что такая очевидно «тривиальная» область математики, как замощение плоскости конгруэнтными «плитками», которая выглядит не более серьезно, чем «детская игра», на самом деле является частью нерекурсивной математики. В действительности эта область содержит множество трудных и не решенных пока задач. Пока неизвестно, например, есть ли единственная «плитка» такой формы, которая бы покрывала всю плоскость непериодически.

[253] В действительности Хао Ванг занимался несколько иной проблемой – с квадратными «плитками», не вращаемыми и с совпадающими по цвету сторонами, – но эти особенности для нас здесь не важны.

[254] Более того, Ханф [1974] и Майерс [1974] показали, что существует отдельное множество (из большого числа «плиток»), которое покрывает плоскость только невычислимым образом.

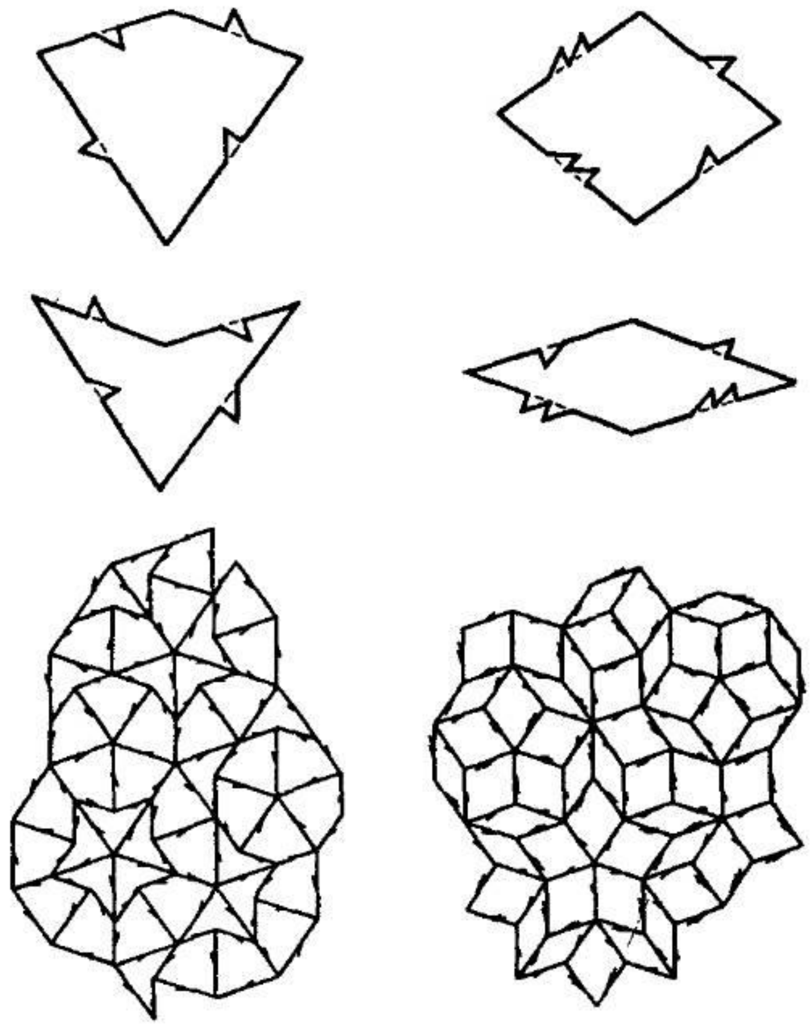


Рис. 4.12. Две пары плиток, которые покрывают плоскость только непериодически («плитки Пенроуза»). Также показано замощение плоскости каждой из этих пар

Задача замощения, в том виде, как она исследовалась Вангом, Бергером и Робинсоном, формулируется для «плиток», построенных на квадратах. Я же здесь допускаю рассмотрение многоугольников произвольной формы, и поэтому необходимо наличие какого-нибудь способа изображения каждой из «плиток», поддающегося адекватному вычислению. Одним из таких путей могло бы быть представление вершин «плиток» точками плоскости Аргана, которые превосходно задаются алгебраическими числами.

§4.10. Похоже ли множество Мандельброта на нерекурсивную математику?

Давайте теперь вернемся к нашей предшествующей дискуссии о множестве Мандельброта. Я буду для наглядности предполагать, что это множество является в некотором смысле нерекурсивным. Поскольку его дополнение рекурсивно нумеруемо, то, как следствие, само оно таковым быть не может. Я думаю, что форма множества Мандельброта может кое-чему научить нас о том, что касается природы нерекурсивных множеств и нерекурсивной математики.

Посмотрим еще раз на рис. 3.2, с которым мы встретились в третьей главе. Заметьте, что бо́льшая часть множества вписывается в сердцевидную фигуру, которую я обозначил на рис. 4.13 через *A*. Эта фигура называется кардиоида, и ее внутренняя область может быть определена математически как множество точек с плоскости Аргана, которые удовлетворяют равенству

$$c = z - z^2$$

где z – комплексное число, чье расстояние до центра координат меньше 1/2. Это множество является, с очевидностью, рекурсивно нумеруемым в смысле существования алгоритма, который для произвольной точки внутренней области фигуры умеет подтверждать ее принадлежность этой самой области. Этот алгоритм легко получается из указанной выше формулы.

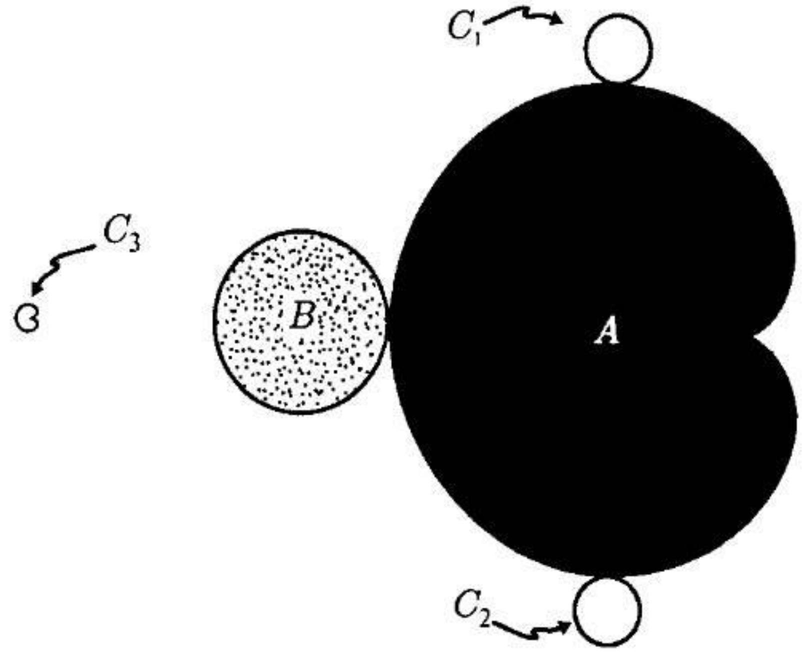


Рис. 4.13. Бóльшая часть внутренней области множества Мандельброта может быть определена простыми алгоритмическими уравнениями

Теперь рассмотрим дисковидную фигуру слева от основной кардиоиды (область B на рис. 4.13). Ее внутренняя часть представляет собой множество точек

$$c = z - 1,$$

где z – удалено от начала координат на расстояние меньше 1/4. Эта область, несомненно, является внутренностью диска, так как представляет собой множество точек, лежащих внутри правильной окружности. И, опять же, эта область является рекурсивно нумеруемой в принятом нами смысле. А как насчет других «бородавок» на кардиоиде? Возьмем две следующие по величине «бородавки». Это практически круглые «кляксы», располагающиеся примерно наверху и внизу кардиоиды на рис. 3.2 и которые на рис. 4.13 обозначены через C_1 и C_2. Они могут быть описаны как множество

$$c^3 + 2c^2 + (1 - z)c + (1 - z)^2$$

где z изменяется в пределах круга радиуса 1/8 с центром в начале координат. Фактически, это уравнение дает нам не только обе эти «кляксы», но и «дочернюю» фигуру диоидной формы (основную часть рис. 3.1), которая находится слева на рис. 3.2 и которая обозначена как C_3 на рис. 4.13. И, аналогично, эти области (как порознь, так и вместе) составляют рекурсивно нумеруемые множества благодаря существованию вышеприведенной формулы.

Несмотря на предположение о нерекурсивности множества Мандельброта, сделанное мной вначале, мы смогли разобраться с его наиболее значительными частями с помощью вполне определенного и достаточно простого алгоритма. Кажется, что такой процесс можно продолжать и дальше. Все наиболее очевидные области множества – и, конечно же, подавляющая часть множества (если не всё оно целиком) в процентном выражении – поддаются алгоритмическому анализу. Если, как я предполагаю, всё множество все-таки нерекурсивно, то те области, которые недоступны для действия алгоритма, должны быть с необходимостью очень «тонкими» и почти «невидимыми». Более того: когда мы найдем такую область, то вероятнее всего мы смогли бы понять, как нам изменить наш алгоритм, чтобы эта область также оказалась в зоне его действия. Однако, после этого найдутся другие области (если мое предположение о нерекурсивности справедливо), еще более труднодоступные из-за тонкости и сложности своей структуры, перед которыми будет бессилен даже наш усовершенствованный алгоритм. И вновь «волшебство» интуиции, искусства и техники,[255] наверное, позволит нам вычленить эту область; но другие в очередной раз ускользнут от нас; и так будет повторяться снова и снова.

Я полагаю, что этот путь не слишком отличается от того, который часто используется в математике для решения трудных и, предположительно, нерекурсивных задач. Многие задачи, с которыми сталкиваются в некоторых специфических областях, часто решаются с помощью простых алгоритмических процедур – процедур, известных, быть может, на протяжении веков. Но некоторые из этих задач могут не поддаться таким методам, и тогда приходится искать более сложные пути к их решению. Такие задачи будут, конечно, сильнее всего интриговать математиков и подталкивать их к развитию всё более мощных методов, в основу которых будет закладываться всё более и более глубокое интуитивное понимание природы используемых математических объектов. Возможно, в этом есть что-то от того, как мы познаем окружающий нас физический мир.

В задачах покрытия и задачах со словами, рассмотренных выше, можно уже уловить, как применяется подобный подход (хотя это не те области математики, где аппарат развит в достаточной степени). Мы смогли привести очень простое доказательство для того, чтобы показать невозможность трансформации одного слова в другое при помощи установленных правил. Нетрудно вообразить, что более «продвинутые» методы доказательства способны помочь в более сложных случаях. Не исключена вероятность, что эти новые подходы могут быть превращены в алгоритмические процедуры. Мы знаем, что ни одна процедура не может удовлетворять всем примерам задачи со словами, но те из них, которые ускользают из «алгоритмических сетей», должны быть очень тонко и аккуратно сконструированы. Конечно, как только мы узнаем принцип построения таких примеров – как только мы будем уверены, что в неком конкретном случае произошла «осечка» алгоритма, – мы сможем усовершенствовать наш алгоритм так, чтобы он включал и этот частный пример. Ускользать могут только пары «неравных» слов, так что, как только мы находим такую «ускользающую» пару, мы можем быть уверены в их «неравенстве» и присовокупить этот критерий к нашему алгоритму. Так наше более глубокое понимание ведет ко всё более совершенным алгоритмам![256]

§4.11. Теория сложности

Рассуждения о природе, возможности построения, существования и ограничениях алгоритмов, которые я привел в предыдущих главах, были по большей части «нестрогими». Я совсем не касался вопроса о возможности практического применения упоминавшихся алгоритмов. Даже в тех задачах, где существование алгоритмов и возможные способы их построения очевидны, всё же может потребоваться довольно много труда для их воплощения в нечто полезное с точки зрения практического использования. Иной раз небольшая догадка или искусный ход могут в значительной степени упростить алгоритм или же многократно увеличить его быстродействие. Техническая сторона этих вопросов часто бывает очень сложна, и в последние годы в различных направлениях прилагалось много усилий в области построения, понимания и совершенствования алгоритмов – быстро растущем и развивающемся поле деятельности для пытливых умов. Мне представляется не слишком уместным углубляться здесь в тонкости подобных вопросов. Однако, существует довольно много абсолютных ограничений общего характера (известных или предполагаемых) на возможное повышение быстродействия алгоритма. Оказывается, что среди алгоритмических по своей природе задач существуют определенные классы проблем, решать которые с помощью алгоритмов несоизмеримо труднее, чем остальные. Такие задачи можно решать только с помощью очень медленных алгоритмов (или, допустим, алгоритмов, требующих чрезмерно больших ресурсов для хранения информации, и т.п.). Теория, в которой рассматриваются подобные вопросы, носит название теории сложности.

Теория сложности занимается не столько изучением трудностей, связанных с решением отдельных задач, сколько с бесконечными семействами задач, в каждом из которых любая задача может быть решена с помощью одного и того же алгоритма. Различные задачи такого семейства будут отличаться по «размеру», который выражается некоторым натуральным числом n. (Чуть

---

[255] **В.Э.:** То есть – другие алгоритмы.

[256] **В.Э.:** Весь процесс эволюции жизни на Земле от первой молекулы ДНК до Роджера Пенроуза есть такой процесс усовершенствования алгоритмов.

позднее я объясню более подробно, как фактически этот номер n характеризует размер задачи.) Время, требуемое для решения конкретной задачи из рассматриваемого класса, – а вернее, количество элементарных шагов, – дается некоторым числом N, зависящим от n. Для определенности договоримся, что N – это наибольшее число шагов среди всех задач данного размера n, которое может понадобиться алгоритму для решения. При этом, с ростом n увеличивается также и N. На самом деле, N скорее всего будет расти гораздо быстрее n. Например, N может быть примерно пропорционально n^2, или n^3, или, скажем, 2^n (которое при больших n значительно превосходит n^2, n^3, n^4, n^5 и, вообще, n^r для любого фиксированного n), или даже 2^{2^n} (которое, в свою очередь, растет еще быстрее).

Конечно, число «шагов» зависит от типа вычислительной машины, на которой применяется алгоритм. Если эта машина принадлежит классу машин Тьюринга, описанному в главе 2, у которых есть только одна лента – что довольно неэффективно – то число N может расти еще быстрее (или, эквивалентно, машина будет работать медленнее), чем в случае с двумя и более лентами. Чтобы избежать этих неопределенностей, вводится широкая классификация всех возможных зависимостей $N(n)$, так что, независимо от типа используемой машины Тьюринга, величина темпов роста N будет всегда попадать в одну и ту же категорию. Одна из таких категорий, известная как **P** (от названия «полиномиальное время»), включает все темпы роста, которые являются фиксированными кратными n или n^2, n^3, n^4, n^5, ...[257] Это означает, что для любой задачи, попадающей в эту категорию **P** (под «задачей» здесь фактически понимается семейство задач, решаемых с помощью единого алгоритма), будет справедлива оценка

$$N \leq K \times n^r$$

где K и r – константы, не зависящие от n. То есть N не может быть больше, чем число, кратное n в некоторой фиксированной степени.

Простой пример задачи, безусловно относящейся к **P**, – перемножение двух чисел. Чтобы объяснить это, я должен сначала описать, как число n характеризует размер двух чисел, которые надо перемножить. Мы можем принять, что оба числа представлены в двоичной записи и что $n/2$ – это просто количество бинарных разрядов в каждом из чисел, так что общее число цифр (то есть битов) у обоих равно n. (Если одно из чисел длиннее другого, то мы можем записать более короткое, начав с дополнительной последовательности нулей, тем самым выровняв их по длине.) Например, если $n = 14$, мы бы могли рассмотреть произведение

$$1011010 \times 0011011,$$

которое является, на самом деле, произведением 1011010 × 11011, но с добавленными перед более коротким числом нулями. Выполнить требуемое действие проще всего путем умножения «в столбик»:

```
        1011010
      × 0011011
      ---------
        1011010
       1011010
      0000000
     1011010
    1011010
   1000000
  0000000
  -------------
  0100101111110
```

учитывая, что в двоичной системе 0 × 0 = 0, 0 × 1 = 0, 1 × 0 = 0, 1 × 1 = 1, 0 + 0 = 0, 0 + 1 = 1, 1 + 0 = 1, 1 + 1 = 10. Число отдельных двоичных перемножений равно $(n/2) \times (n/2) = n^2/4$, а число отдельных двоичных сложений может доходить до $n^2/4 - n/2$ (включая перенос). Это дает $n^2/2 - n/2$ отдельных арифметических операций – и мы должны еще учесть несколько дополнительных логических шагов, которые задействованы в операциях переноса. Тогда общее число шагов,

[257] Понятие «полиномиальный» применяется на самом деле к выражениям более общего вида, скажем, $7n^4 - 3n^3 + 6n + 15$, но это никак не изменит общности наших рассуждений. В любом таком выражении все члены младших степеней n теряют значимость по мере увеличения n (так что в данном примере можно игнорировать все слагаемые, кроме $7n^4$).

игнорируя члены более низкого порядка, равно по существу $N = n^2/2$, что, очевидно, является полиномом.[258]

В общем случае, мы полагаем «размер» n задачи из некоторого класса равным полному количеству двоичных цифр (или битов), необходимых для задания свободных входных данных в задаче указанного размера. Другими словами, для произвольного размера n задача может иметь до 2^n различных вариантов (ибо для каждой из цифр имеется две возможности – 0 или 1, – а общее количество цифр равно n), и все они должны одинаково обрабатываться алгоритмом не более, чем за N шагов.

Существует масса примеров (классов) задач, которые не «принадлежат» множеству P. Например, чтобы вычислить 2^{2^r} для заданного натурального r, нам только для <u>записи конечного ответа</u> потребуется около 2^n шагов (где n – число цифр в двоичной записи r), не говоря даже о самом вычислении. Операция по вычислению $2^{2^{2^r}}$ потребует уже 2^{2^n} шагов для записи и так далее. Значения этих выражений намного превосходят те, которые дают полиномы для тех же n, и, следовательно, не могут принадлежать **P**.

Больший интерес представляют задачи, в которых ответ может быть записан и даже проверен на верность за «полиномиальное» время. Есть очень важная категория (алгоритмически решаемых классов) проблем, обладающих таким свойством. Их называют **NP**-задачами (классом задач). Точнее, если некоторая задача из класса **NP** имеет решение, то алгоритм позволит получить это решение, которое затем может быть проверено за «полиномиальное» время. Если же задача не имеет решения, то алгоритм сообщит об этом, но при этом не оговаривается необходимость проверки этого факта за «полиномиальное» или какое бы то ни было время.[259]

NP-задачи встречаются во многих областях, причем как в математике, так и в повседневной практике. Я приведу здесь только один простой математический пример: задачу нахождения так называемого «<u>гамильтонова цикла</u>» на графе (довольно устрашающее название для чрезвычайно простой идеи). Под графом подразумевается конечный набор точек, или «вершин», некоторое количество пар которых соединено между собой линиями – «сторонами» графа. (Нас не интересуют сейчас геометрические или линейные свойства, а только то, какие вершины соединяются друг с другом. Поэтому не имеет значения, лежат ли все вершины в одной плоскости – если нас не волнует возможность пересечения двух сторон – или же в трехмерном пространстве.) Гамильтонов цикл – это замкнутый маршрут (петля), состоящий только из сторон графа и проходящий не более одного раза через любую из вершин. Пример графа с изображенным на нем гамильтоновым циклом показан на рис. 4.14. Задача нахождения гамильтонова цикла заключается в том, чтобы определить, существует ли гамильтонов цикл на рассматриваемом графе, и если существует, то явным образом указать его.

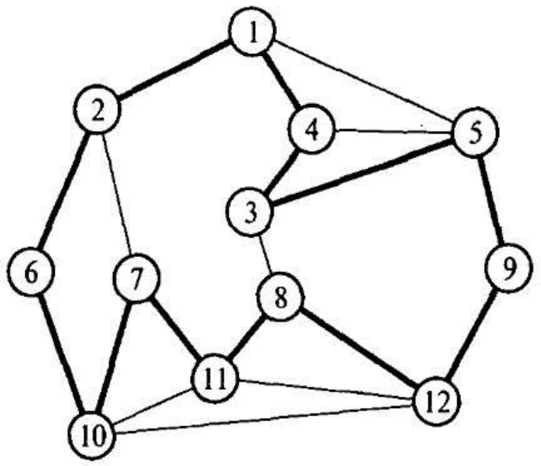


Рис. 4.14. Граф с гамильтоновым циклом (изображен зачерненными линиями). Существует только один гамильтонов цикл, как читатель может сам убедиться

Есть разные способы представления графов на языке двоичных чисел. Неважно, какой из этих способов применяется в том или ином случае. Один из методов заключается в том, чтобы пронумеровать вершины 1, 2, 3, 4, 5, ... , а потом перечислить пары в некотором подходящем фиксированном порядке:

$$(1, 2), (1, 3), (2, 3), (1, 4), (2, 4), (3, 4), (1, 5), (2, 5), (3, 5), (4, 5), (1, 6), \ldots .$$

[258] В действительности, путем применения некоторых тонких ходов, можно сократить число шагов до величины порядка

$$n \log n \log (\log n)$$

для больших n – которая, конечно, всё еще принадлежит P. За подробностями я отсылаю читателя к Кнуту [1981].

[259] Если быть точным, классы **P**, **NP** и **NP**-полный (см. с. 123) определены только для задач типа «да или нет» (скажем, когда заданы a, b, c и спрашивается, выполняется ли для них $a \times b = c$); но описания, приведенные в тексте, вполне подходят для наших целей.

Затем мы на место каждой пары помещаем «1», если пара соединена стороной графа, и «0» – в противном случае. Тогда двоичная последовательность

10010110110...

будет означать, что вершина 1 соединяется с вершинами 2, 4 и 5; вершина 3 с вершинами 4 и 5; вершина 4 – с вершиной 5, и т.д. (в соответствии с рис. 4.14). Гамильтонов цикл может быть задан по желанию просто как подмножество этих сторон, которое было бы описано такой же двоичной последовательностью, как и ранее, но со значительно бо́льшим числом нулей. Процедура проверки в этом случае проходит несравненно быстрее, чем процесс непосредственного построения гамильтонова цикла. Всё, что нужно выяснить, – это является ли построенный цикл действительно циклом, т.е. принадлежат ли его стороны исходному графу, и что каждая вершина графа используется ровно два раза – по одному разу на концах каждой из входящих в нее двух сторон. Такую процедуру проверки можно легко завершить за «полиномиальное» время.

На самом деле эта задача относится не только к **NP**, но к так называемой категории **NP**-полных задач. Это означает, что любая другая **NP**-задача может быть сведена к данной за «полиномиальное» время – так что, если бы кому-нибудь удалось отыскать алгоритм для решения задачи нахождения гамильтонова цикла за «полиномиальное» время (т.е. показать, что задача гамильтонова цикла действительно принадлежит **P**), то это будет означать, что все **NP**-задачи будут лежать в **P**! Это имело бы очень важные следствия. В широком смысле, задачи из **P** считаются «податливыми» (иначе говоря, «решаемыми за приемлемое время») для относительно больших n, на быстром современном компьютере; тогда как задачи из **NP**, но не лежащие в **P**, считаются «неподатливыми» (т.е. решаемыми в принципе, но «нерешаемыми практически») для тех же n – независимо от того, на какое разумно предсказуемое увеличение быстродействия компьютеров рассчитывать в будущем. (Реальное время, которое бы потребовалось для достаточно больших n при решении «неподатливой» задачи, легко превосходит возраст вселенной, что никак не предполагает практическое использование такого подхода!) Любой «умный» алгоритм для решения задачи о нахождении гамильтонова цикла за «полиномиальное» время мог бы быть превращен в алгоритм для решения всех прочих **NP**-задач, и тоже за «полиномиальное» время!

Другая задача, также являющаяся **NP**-полной[260] – «задача коммивояжера», которая во многом похожа на гамильтонов цикл, если не считать того, что разным сторонам приписаны числа и ставится цель отыскать гамильтонов цикл с минимальной суммой этих чисел (минимальной «длиной» пути, проделанного коммивояжером). Аналогично, «полиномиальное» время решения, достигнутое в «задаче коммивояжера», привело бы к возможности решать все **NP**-задачи за «полиномиальное» время. (Если такое решение когда-нибудь найдется, то новость об этом сразу попала бы на первые страницы! Ведь к **NP**-задачам относится, в частности, факторизация больших целых чисел, которая применяется в секретных шифровальных системах, представленных за последние несколько лет. Если эта задача окажется решаемой за «полиномиальное» время, то, возможно, такие шифры могли бы быть взломаны при помощи мощных современных компьютеров; если же нет – эти шифры останутся неприступными. См. Гарднер [1989].)

Эксперты, как правило, полагают, что используя устройство, работающее по принципу машины Тьюринга, невозможно за «полиномиальное» время решить **NP**-полную задачу; и что, следовательно, **P** и **NP** – неэквивалентны. Это мнение, похоже, верно, хотя пока его никто не смог доказать. И это остается наиболее важной и на сегодняшний день нерешенной задачей теории сложности.

§4.12. Сложность и вычислимость в физических объектах

Теория сложности является важной для наших рассуждений в этой книге не только потому, что она касается вопроса возможности алгоритмизации, но и потому, что она позволяет для заведомо алгоритмизуемых объектов решать вопрос о том, могут ли использоваться соответствующие алгоритмы на практике. В последующих главах я буду больше говорить о вычисли-

---

[260] Строго говоря, нам нужно переформулировать эту задачу под ответ «да или нет», например: существует ли маршрут для коммивояжера, длина которого меньше чем столько-то? (См. предыдущее примечание.)

мости, чем о теории сложности, поскольку я склонен думать (хотя, конечно, и не имея для этого достаточных оснований), что, в отличие от фундаментального вопроса вычислимости, положения теории сложности не настолько значимы для феномена мышления. Более того, мне представляется, что теория сложности сегодня лишь слегка затрагивает вопросы практичности алгоритмов.

Однако, я могу кардинально ошибаться по поводу важности той роли, которую играет сложность. Как будет показано позднее (глава 9, {с.323}), теория сложности для реальных физических объектов, вероятно, может существенно отличаться от теории, изложенной мной ранее. Чтобы с уверенностью констатировать эту возможную разницу, необходимо будет использовать некоторые волшебные свойства квантовой механики – мистической, но всё же поразительно точной теории, описывающей поведение атомов и молекул, а также и другие явления, многие из которых представляют интерес и на макромасштабах. Мы познакомимся с этой теорией в главе {6}. Согласно ряду идей, предложенных Давидом Дойчем [1985], существует принципиальная возможность построить «квантовый компьютер», на котором за «полиномиальное» время могут быть решены некоторые задачи (или классы задач), не принадлежащих **P**. Пока совершенно неясно, как на практике сконструировать такое физическое устройство, которое бы (надежно) функционировало по принципу «квантового компьютера» – и, более того, рассматриваемый до сих пор класс задач носил заведомо искусственный характер, – но теоретически понятно, что квантовое физическое устройство могло бы улучшить работу машины Тьюринга.

А есть ли вероятность, что человеческий мозг, который в рамках данного обсуждения я рассматриваю как физическое устройство, хотя и имеющее чрезвычайно тонкую и сложную структуру – может неким образом использовать волшебство квантовой теории? Понимаем ли мы сегодня, как именно квантовые эффекты могут с пользой применяться для решения задач или формирования суждений? Можем ли мы представить, что для использования этих возможных преимуществ нам придется выйти «за нынешние пределы» квантовой теории? Насколько вероятно усовершенствование реальных физических устройств с учетом теории сложности для машин Тьюринга? И что говорит о таких устройствах теория вычислимости?

Чтобы рассматривать эти вопросы, нам надо будет отойти на время от математических абстракций и задаться целью выяснить в следующих главах, как же, в действительности, ведет себя окружающий нас мир!

(Продолжение в файле {PENRO3 = МОИ № 15})

Научно-популярное издание
«Мысли об Истине»
Выпуск № 14
Сформирован 28 июля 2014 года

Все читатели приглашаются принять участие в создании альманаха МОИ и присылать свои статьи и заметки для этого издания по адресу: Marina.Olegovna@gmail.com. Если присланные материалы будут соответствовать направлению Альманаха и минимальным требованиям информативности и корректности, то они будут опубликованы в нашем издании.

Основной вид существования Альманаха МОИ – в виде PDF-файлов в Вашем компьютере. Держите все выпуски МОИ в одной папке. Скачать PDF-ы можно с разных мест в Интернете, и не важно, откуда номер скачан. В Интернете нет одной фиксированной резиденции МОИ.

Содержание

Предисловие к пяти выпускам альманаха МОИ 2
Файл PENRO1 3
Предисловие в Векордии 3
Роджер Пенроуз. «Новый Разум Короля» 4
Обращение к читателю 4
Как читать математические формулы 4
Благодарности 4
Предисловие 5
Вступление 8
Пролог 17
Глава 1. Может ли компьютер обладать разумом? 18
§1.1. Введение 18
§1.2. Тест Тьюринга 20
§1.3. Искусственный интеллект 31
§1.4. Подход к понятиям «удовольствия» и «боли» с позиций ИИ 34
§1.5. Сильный ИИ и китайская комната Серла 36
§1.6. «Железо» и «софт» 47
Глава 2. Алгоритмы и машины Тьюринга 52
§2.1. Основы алгоритмов 52
§2.2. Концепция Тьюринга 56
§2.3. Двоичная запись цифровых данных 62
§2.4. Тезис Черча–Тьюринга 66
§2.5. Числа, отличные от натуральных 68
§2.6. Универсальная машина Тьюринга 69
§2.7. Неразрешимость проблемы Гильберта 75
§2.8. Как превзойти алгоритм 81
§2.9. Лямбда-исчисление Черча 84
Файл PENRO2 92
Предисловие в Векордии 92
1. Начало математики 93
2. Классификация множеств программой N 93
3. Конкретные множества 94
4. Абстрактные множества 95
5. Классификация отношений множеств программой R 97
6. Первичные и вторичные алгоритмы 98
7. Термины Веданской теории, применяемые в комментариях 99
Роджер Пенроуз. «Новый Разум Короля» 101
Глава 3. Математика и действительность 101
§3.1. Страна Тор'Блед-Нам 101
§3.2. Действительные числа 103
§3.3. Сколько же всего действительных чисел? 107
§3.4. «Действительность» действительных чисел 111
§3.5. Комплексные числа 112
§3.6. Построение множества Мандельброта 115
§3.7. Платоническая реальность математических понятий? 117
Глава 4. Истина, доказательство и интуиция 120
§4.1. Программа Гильберта для математики 120
§4.2. Формальные математические системы 123
§4.3. Теорема Гёделя 126
§4.4. Математическая интуиция 128

§4.5. Платонизм или интуиционизм? 132
§4.6. Теоремы гёделевского типа как следствие результатов, полученных Тьюрингом 135
§4.7. Рекурсивно нумеруемые множества 137
§4.8. Является ли множество Мандельброта рекурсивным? 141
§4.9. Некоторые примеры нерекурсивной математики 144
§4.10. Похоже ли множество Мандельброта на нерекурсивную математику? 151
§4.11. Теория сложности 153
§4.12. Сложность и вычислимость в физических объектах 156
Содержание 158