Недетерминированная вычислимость: философские основания[1]

А. М. Анисов

ABSTRACT. The paper consists of three parts. The first one dwells on a comparison of deterministic and nondeterministic computability and defines two kinds of nondeterministic computabilities: global and local. The second part considers the sources of nondeterministic computations. The third part touches upon the problems of computational modelling of causal relationships.

Ключевые слова: детерминированная вычислимость, недетерминированная вычислимость, линейная дискретность, глобальный недетерминизм, локальный недетерминизм, оракулы, производительная причинность.

1 Детерминированная и недетерминированная вычислимость

Детерминированная вычислимость ограничена вычислением *функций*. В теориях вычислимости применяется несколько отличное от обычного теоретико-множественного понятие функции. Зафиксируем произвольное непустое множество U в качестве универсума. Любое подмножество F^n множества U^n (здесь U^n есть n-кратное ($n \geq 1$) декартово произведение множества U, т.е. множество всех упорядоченных n-ок элементов из U) называется *n-местным отношением* на U.

В том случае, если $n+1$-*местное отношение* F^{n+1} на U удовлетворяет условию

$(\forall < x_1, x_2, \ldots, x_n, x_{n+1} > \in F^{n+1})(\forall < y_1, y_2, \ldots, y_n, y_{n+1} > \in F^{n+1})$

$((< x_1, x_2, \ldots, x_n > = < y_1, y_2, \ldots, y_n >) \rightarrow (x_{n+1} = y_{n+1}))$,

F^{n+1} называется *n-местной функцией* на U.

[1]Работа выполнена при поддержке РГНФ, грант №07-03-00203а.

Функция F^{n+1} является *тотальной* на U, если $F^{n+1} = U \times U \times \cdots \times U \times P$, где $P \neq \varnothing$ и $P \subset U$. Функция F^{n+1} будет *частичной* на U, если $F^{n+1} = F_1 \times F_2 \times \cdots \times F_n \times P$, где $P \subset U$, для всех F_j $(1 \leq j \leq n)$ $F_j \subset U$ и для хотя бы одного F_j $(1 \leq j \leq n)$ имеет место $F_j \neq U$. Ясно, что каждая функция на U будет либо тотальной, либо частичной. Содержательно, тотальная функция имеет значение x_{n+1} для любой n-ки $< x_1, x_2, \ldots, x_n > \in U^n$, тогда как частичная функция хотя бы для одной такой n-ки не определена. Выделенным случаем частичной функции является пустая функция $\varnothing$.

В теории множеств нет нужды различать понятия тотальной и частичной функции. При фиксированном U тотальная и частичная функции на U будут просто функциями с разными областями определения. Но в теории вычислимости выделение данных понятий имеет важное значение. Вычислимой тотальной функции соответствует процедура, заканчивающаяся на любом входе, тогда как вычислимой частичной функции сопоставляется процедура, процесс выполнения которой на некоторых входах не имеет последнего шага выполнения. Как известно, проблема останова является неразрешимой: доказано, что не существует алгоритма, отделяющего программы, вычисляющие тотальные функции, от программ, вычисляющих час тичные функции. Поэтому разница между тотальными и частичными вычислимыми функциями становится принципиальной.

Следуя идеям Марио Бунге, выдвинутым в книге [5], определим **детерминизм** как принцип, согласно которому все существующее, во-первых, чем-то *однозначно обусловлено*, и, во-вторых, во всем *полностью определено*. Отсюда **недетерминизм** означает либо неоднозначную обусловленность, либо неполную определенность, или и то и другое вместе. Крайней формой недетерминизма является **индетерминизм** — допущение существования чего-либо ничем не обусловленного или полностью неопределенного.

В работе [1] было показано, что классическая трактовка вычислимости связана с тремя типами недетерминизма. Во-первых, в ней принимается недетерминизм в «мягкой» форме неопределенности итогового результата вычислений и случайности появления начальных данных. Во-вторых, допускается «жесткий»

недетерминизм в виде настоящего элемента индетерминизма, имеющего место в силу неопределенности и необусловленности процесса получения начальных данных без использования случайного процесса. В-третьих, если бы теоретические вычислительные устройства были воплощены в материи, то им был бы присущ еще один тип недетерминизма, связанный с возможностью неправильной работы: такие компьютеры могли бы завершать вычисления раньше времени, зависать, неверно выполнять команды и т.д. Однако в теориях классической вычислимости ничего подобного не может случиться в принципе.

Но если отвлечься от этих форм недетерминизма, классически понимаемая вычислимость (машины Тьюринга, МНР-компьютеры, нормальные алгорифмы Маркова и т.д.) оказывается полностью детерминированной в следующем смысле. Если введены начальные или промежуточные данные, результат следующего шага по их обработке оказывается однозначным. Тем самым классические вычисления, во-первых, полностью обусловлены предшествующими данными и, во-вторых, полностью предопределены в отношении последующих результатов. Это остается верным, даже если конкретный вычислительный процесс никогда не завершается: все равно каждый шаг вычислений и в такой ситуации является обусловленным и определенным. Речь идет, таким образом, не о глобальной, а о *локальной* детерминированности. В целом же не завершающийся вычислительный процесс следует признать *глобально* недетерминированным, поскольку отсутствует определенный результат такого вычисления.

Сказанное требует уточнения. Делать это можно по-разному, но мы будем исходить из следующих положений. Всякое вычисление предполагает наличие допустимых для обработки данных и соответствующих преобразующих эти данные *объектных* операций или команд. Могут также иметься *управляющие* операции или команды, определяющие порядок применения (или не применения) других операций в ходе вычислений и не затрагивающие обрабатываемые данные. В любом случае результатом применения команды будет наступление некоторого *события*. Назовем **вычислительным процессом** (или, короче, **вычислением**) *любое реализуемое в рамках некоторой теории вычислений линейное дискретное множество событий.*

По определению, *линейное дискретное множество* — это множество M, упорядоченное отношением R, удовлетворяющим следующим аксиомам.

1. $\forall x \forall y \neg(xRy)$ — антирефлексивность.
2. $\forall x \forall y \forall z((xRy \& yRz) \to xRz)$ — транзитивность.
3. $\forall x \forall y(xRy \vee yRx \vee x = y)$ — линейность.
4. $\forall x \forall y(xRy \to \exists z(xRz \& (zRy \vee z = y) \& \neg \exists v(xRv \& vRz)))$ — дискретность справа.
5. $\forall x \forall y(xRy \to \exists z(zRy \& (xRz \vee z = x) \& \neg \exists v(zRv \& vRy)))$ — дискретность слева.

Кванторы ограничены множеством M: $\forall x$ распишется как $\forall x(x \in M \to$, а $\exists x$ как $\exists x(x \in M \&$. Поэтому пустое множество и любой синглетон (т.е. одноэлементное множество) будут вырожденными примерами линейных дискретных множеств (при этом будем иметь $R = \varnothing$).

В теориях классической вычислимости каждый вычислительный процесс может быть представлен либо в виде конечной последовательности (возможно, повторяющихся) событий

$S_1, S_2, \ldots, S_n,$

либо в виде бесконечной последовательности (не исключено, повторяющихся конечное или бесконечное число раз) событий

$S_1, S_2, \ldots, S_n, \ldots$.

Очевидно, что такие последовательности событий являются простейшими примерами линейных дискретных множеств. Кроме того, они *реализуемы* в рамках классической теории вычислимости. Конечные последовательности соответствуют здесь заканчивающемуся на данном входе или *сходящемуся* вычислению, тогда как бесконечная последовательность представляет никогда не завершающееся или *расходящееся* вычисление. Каждый переход $S_S \longmapsto S_{S+1}$ полностью детерминирован, так что все классические вычисления локально детерминированы. Расходящиеся вычисления глобально недетерминированы в силу неопределенности итогового результата. Кстати говоря, сложившееся понятие алгоритма как полностью детерминированного

предписания по однозначному решению задач исключает возможность не останавливающихся вычислений.

В получивших известность обобщениях теории классической вычислимости и теориях неклассической вычислимости не могут быть реализованы вычислительные процессы, наглядно представленные следующими линейно дискретно упорядоченными множествами событий:

$\ldots, S_{-n}, \ldots, S_{-2}, S_{-1},$

$\ldots, S_{-n}, \ldots, S_{-2}, S_{-1}, S_0, S_1, S_2, \ldots, S_n, \ldots$.

Этим множествам соответствуют вычислительные процессы, не имеющие первого шага выполнения. Еще одним примером классически не реализуемого процесса будет множество событий, упорядоченное по типу ординала $\omega + 1$:

$S_1, S_2, \ldots, S_n, \ldots S_\omega$.

Зато в некоторых неклассических теориях принимается так называемое ω-правило, позволяющее от бесконечной ω-последовательности посылок $\varphi(1), \varphi(2), \ldots, \varphi(n), \ldots$ на $\omega+1$-шаге переходить к заключению $\forall x \varphi(x)$. Более того, в теории α-рекурсии (см. [16]) допускается вычислимость до любых подходящих ординалов, превышающих ординал ω. Однако в нашем смысле трансфинитная ω + 1-последовательность событий $S_1, S_2, \ldots, S_n, \ldots S_\omega$, не говоря уже о трансфинитных α-последовательностях, в которых $\alpha > \omega + 1$, просто не будет представлением никаких вычислительных процессов. В самом деле, в таких последовательностях нарушается аксиома 5 дискретности слева, согласно которой каждый элемент, имеющий предшественника, имеет и непосредственного предшественника. Но уже первый выход за пределы ординала ω приводит к тому, что у элемента S_ω предшественники есть, а непосредственного соседа слева нет.

Отметим, что для α-последовательностей аксиома 4 дискретности справа выполняется. Обратная ситуация, когда 5 выполняется, а 4 нет, возникнет, например, в следующем случае. Рассмотрим множество событий

$S_{\omega *} \ldots, S_{-n}, \ldots, S_{-2}, S_{-1}$.

(Здесь $\omega *$ является порядковым типом, соответствующим стандартному порядку на множестве отрицательных чисел.)

Такое множество имеет первый и последний элементы, однако за первым событием $S_{\omega *}$ непосредственно следующего нет,

что обусловливает нарушение аксиомы 4. Но каждый имеющий предшественника элемент имеет и непосредственного предшественника, так что аксиома 5 выполняется.

Наконец, объединяя последние две ситуации, получим множество событий

$$S_{\omega *} \ldots, S_{-n}, \ldots, S_{-2}, S_{-1}, S_1, S_2, \ldots, S_n, \ldots S_\omega,$$

которое не будет дискретным множеством по причине нарушения обеих аксиом 4 и 5. И вновь таким множествам, согласно принятому определению, не соответствуют никакие вычислительные процессы.

Против упомянутого определения вычислительного процесса напрашиваются два принципиальных возражения. Во-первых, требование линейности исключает рассмотрение ветвящихся процессов. Между тем, именно такие процессы являются одним из главных видов недетерминированных вычислений. Во-вторых, аксиомы дискретности блокируют идею непрерывных процессов, которая хорошо укладывается в материал естествознания, в первую очередь физики. Кроме того, блокируется возможность обобщения теории вычислимости на подходящие трансфинитные ординалы, превышающие ординал ω.

Уточнение идеи вычислительного процесса основывалось на определенных философских представлениях. Дискретность вычислений позволяет нам подключить интуицию процессуальности, ибо все, что требуется от вычислительного процесса с незамутненной интуитивной точки зрения, — это возможность разложения его на отдельные шаги, причем на каждом шаге (если этот шаг не первый и не последний в серии процессуальных актов) можно указать непосредственно предшествующий и непосредственно следующий шаги. Те, у кого интуиция иная, обязаны этим обстоятельством вошедшим в привычку искусственным желанием втиснуть реальный мир в модели, существенным образом использующие идею непрерывности или, по крайней мере, идею плотности.

Наиболее ярко это проявляется в господствующих в точном естествознании моделях времени. В них моменты времени отождествляются с действительными числами, а временн*о*е отношение «раньше, чем» — со стандартным порядком «меньше, чем» на таких числах. Как известно, данный порядок является ли-

нейным и непрерывным. Последнее свойство влечет *плотность* данного порядка, т.е. выполняется формула

$\forall x \forall y(x < y \rightarrow \exists z(x < z \& z < y \& z \neq x \& z \neq y))$.

Обычным образом упорядоченное множество рациональных чисел оказывается плотным, хотя и не является непрерывным. Но этого достаточно, чтобы покончить с идеей осуществляющегося шаг за шагом процесса. Ведь свойство плотности несовместимо с дискретностью. Рассмотренные нарушающие дискретность структуры $S_1, S_2, \ldots, S_n, \ldots S_\omega$ и $S_{\omega*} \ldots, S_{-n}, \ldots, S_{-2}, S_{-1}$ осуществляют это нарушение за счет следующего уподобления свойству плотности:

$\forall S_j(S_j R S_\omega \rightarrow \exists S_k(S_j R S_k \& S_k R S_\omega \& S_k \neq S_j \& S_k \neq S_\omega))$,

$\forall S_j(S_{\omega*} R S_j \rightarrow \exists S_k(S_{\omega*} R S_k \& S_k R S_j \& S_k \neq S_j \& S_k \neq S_{\omega*}))$.

Между тем, никем, никогда и нигде не было приведено ни одного примера реально существующей плотной структуры. Напротив, чем больше мы узнаем об окружающей действительности, тем больше на первый план выходит идея дискретности природных процессов. Достаточно в связи с этим упомянуть квантовую физику. Но и первоначально, до появления новоевропейского естествознания, в науке и в ориентированной на науку философии преобладало представление о дискретности реальности. Затем восприятие реальности было искажено возобладавшими в науке (особенно в физике) описаниями действительности средствами непрерывных моделей. Такое развитие событий не было случайностью. Недаром закончилась ничем попытка античных атомистов создать атомистическую математику. Теперь стало ясным, что дискретное моделирование реальности с логической и математической точек зрения идейно и технически оказалось более сложным, чем применение в этих же целях привычного и хорошо разработанного аппарата классической математики. (Представление о возникающих тут проблемах можно получить, обратившись, например, к книге [13]).

Самым знаменитым проявлением парадоксальности идеи дискретного процесса явились появившиеся еще на заре философии и науки апории Зенона Элейского о движении. Зенон не сомневается в том, что движение не может быть ничем иным, как некой дискретной последовательностью актов или шагов. Но апории говорят о том, что при всех возможных вариантах

описания движения в виде такой последовательности возникают неустранимые противоречия. Отсюда вывод — движения нет. Нововременная наука справилась с описанием движения ценой отказа от идеи его дискретности. Тем самым в жертву была принесена естественная интуиция дискретности процесса движения. Но самое удивительное в том, что в массовом масштабе воспитанные в традиционной парадигме физики просто не видят здесь проблемы. Стандартное образование ломает естественную интуицию! Апории Зенона для них — не более, чем остроумные, но неверные донаучные рассуждения или даже софизмы. Этот круг вопросов подробно рассматривается в книге [2].

Мы попытались кратко ответить на второе принципиальное возражение, касающееся дискретности. Теперь обратимся к первому, связанному с линейностью. Вновь в основе принятия идеи линейности лежат философские соображения. Представим себе житейскую ситуацию: человек принимает решение, жениться ему или не жениться. В пользу того или иного решения он может привести серьезные в его глазах доводы. Предположим также, что любое решение реально осуществимо. Означает ли это, что описанная ситуация может быть представлена ветвящимся процессом? Если принять во внимание наличествующие *возможности*, то «да». Если же имеется в виду лишь то, что *реализуемо*, то «нет». Эти две возможности исключают совместную реализацию. Реализация потребует сделать выбор в пользу лишь одного решения.

Обозначим этап раздумий посредством α, через β — событие женитьбы, через $\neg\beta$ — событие отказа от женитьбы. Стрелки $\nearrow$ и $\searrow$ выражают элементарный одношаговый процесс — акт перехода от события к событию. Представим утвердительный ответ схемой $\alpha\begin{array}{l}\nearrow\beta\\\searrow\neg\beta\end{array}$. Не только по философским соображениям, но и лингвистически неудачно называть эту схему единым процессом. На самом деле мы вольно или невольно выделяем *два* далее неразложимых элементарных возможных процесса: $\alpha \longrightarrow \beta$ и $\alpha \longrightarrow \neg\beta$, которые к тому же несовместимы между собой по итоговому результату. Люди так и рассуждают, когда говорят о будущих альтернативах: возможно как β, так и $\neg\beta$, но не то и другое вместе.

В отношении ветвления процессов в прошлое вывод еще более

категоричен: такого не следует допускать даже в возможности, так как ничто в реальности схеме, подобной $\overset{\beta}{\underset{\gamma}{\searrow\!\nearrow}}\alpha$, не соответствует. Не может быть единой истории вычислений или единой истории чего-угодно с двумя различными прошлыми. Допустим, вычислено, что эквиваленция $A \leftrightarrow B$ истинна (событие α). Это означает, что прежде имело место либо A и B одновременно истинны (событие β), либо A и B одновременно ложны (событие γ). Но это, конечно, не означает, что в прошлом события β и γ реализовались одновременно. Было либо одно, либо другое, но что-то одно.

Таким образом, *актуально* реализующих ветвление в прошлое или в будущее процессов не бывает. Также есть все основания ввести ветвление будущих возможностей, относя каждую возможность при каждом пункте ветвления к отдельному процессу (в примере возможен как процесс женитьбы, так и процесс отказа от женитьбы). Но ветвлений процессов в прошлое быть не может ни в каком смысле. Разными могут быть лишь наши представления об одном и том же прошлом.

Высказанные соображения приводят к мысли о том, что отдельные вычислительные процессы сами по себе мало что могут дать для экспликации идеи недетерминизма. Рассматривая процессы $\alpha \longrightarrow \beta$ и $\alpha \longrightarrow \neg\beta$ по отдельности, мы не сможем увидеть проявление недетерминизма. Соединяя их единой схемой $\alpha\overset{\beta}{\underset{\neg\beta}{\nearrow\!\searrow}}$, попадаем, как было показано, в философски неприемлемую ситуацию. Но выход есть. Он состоит в переходе от рассмотрения отдельных процессов к анализу *множеств процессов*. В используемом примере вместо двух отдельных процессов и вместо объединяющей их схемы можно предложить теоретико-множественную пару $\{\alpha \longrightarrow \beta, \alpha \longrightarrow \neg\beta\}$. Однако пока это сугубо формальный прием. Необходимо придать ему смысл.

Для этого введем понятие программы. **Программой** *будем называть непустую конечную последовательность команд.* Выполнение программы π на компьютере @ порождает некоторый вычислительный процесс α. Возможна ситуация, когда $\alpha = \varnothing$. Это произойдет в том случае, если ни одна из команд программы π не была выполнена компьютером @ либо по причине его маломощности, либо из-за логической невозможности выполне-

ния. Ассоциируем с каждой парой $(\pi, @)$ *множество* $\Theta_\pi^@$ *всех вычислительных процессов, которые могут быть порождены компьютером* $@$ *при выполнении программы* π.

Отметим следующий факт.

ТЕОРЕМА 1. *Для любых* π *и* $@$ *множество* $\Theta_\pi^@$ *не пусто:* $\Theta_\pi^@ \neq \varnothing$.

Доказательство. Если в результате выполнения программы π компьютером $@$ будет выполнена одна или более команд, возникнет непустой вычислительный процесс $\alpha \neq \varnothing$, $\alpha \in \Theta_\pi^@$. Если же никаких команд при попытке выполнить π выполнено не будет, получим пустой вычислительный процесс $\alpha = \varnothing$, $\alpha \in \Theta_\pi^@$.

Q.E.D.

Введем следующие обозначения. Запись $\downarrow \alpha$ указывает, что процесс α имеет первый шаг выполнения. Формула $\uparrow \alpha$ означает, что процесс α не имеет начала. Аналогичным образом, записи $\alpha \downarrow$ и $\alpha \uparrow$ указывают, соответственно, на наличие последнего шага выполнения для α и отсутствие последнего шага для α. Возникают четыре комбинации стрелок, имеющие очевидный смысл: $\downarrow \alpha \downarrow$, $\downarrow \alpha \uparrow$, $\uparrow \alpha \downarrow$, $\uparrow \alpha \uparrow$.

Но есть еще одна проблема — проблема *аварийного останова* или, короче, *авоста*. Имеется в виду не техническая проблема поломки компьютера, а *логический авост*. Процесс α может логически «зависнуть», когда реализующий его компьютер, выполнив непустой ряд команд, не сможет выполнить очередную команду из-за логической невозможности ее выполнить. В этом случае процесс α не пуст, но в то же время не подпадает ни под одну из перечисленных разновидностей процессов. То же самое касается и пустого процесса: ведь при $\alpha = \varnothing$ нельзя сказать ни того, что процесс α начался и закончился (каким событием начался и каким закончился?), ни того, что он никогда не начинался и никогда не закончится (что предполагает бесконечный в обе стороны ряд событий, которых здесь заведомо нет). Будем применять запись $\alpha = \beta\varnothing$ в знак того, что процесс α успешно выполнялся на имеющем последнее событие S_j подпроцессе β (т.е.имело место $\beta \downarrow$), но при вычислении следующего события S_{j+1} возник логический авост. Если же такого успешного этапа β

не было и сразу наступил авост, просто имеем $\alpha = \varnothing$. Формально есть еще запись вида $\varnothing\beta$. Но это бессмысленная формула. Логическое «зависание» не может быть для чего-то началом, не может породить выполнение каких бы то ни было команд и соответствующей цепочки событий. Оно всегда фатально и безысходно.

В итоге получаем следующие семь разновидностей вычислительных процессов:

$\downarrow \alpha \downarrow$, $\downarrow \alpha \uparrow$, $\uparrow \alpha \downarrow$, $\uparrow \alpha \uparrow$, $\downarrow \alpha\varnothing$, $\uparrow \alpha\varnothing$, $\varnothing$.

Лишь вычислительные процессы типа $\downarrow \alpha \downarrow$ могут быть отнесены к детерминированным, точнее, *глобально детерминированным*. Вычислительные процессы остальных шести типов будут *глобально недетерминированными* из-за возникающих в связи с ними неопределенностей.

Это дает мотивировку для принятия следующего определения. *Множество вычислительных процессов* $\Theta^{@}_{\pi}$ **глобально детерминировано**, *если* $\forall\alpha(\alpha \in \Theta^{@}_{\pi} \rightarrow\downarrow \alpha \downarrow)$. Поскольку по теореме $\Theta^{@}_{\pi} \neq \varnothing$, отсюда вытекает, что хотя бы один процесс $\alpha \in \Theta^{@}_{\pi}$ и при этом $\downarrow \alpha \downarrow$. *Множество вычислительных процессов* $\Theta^{@}_{\pi}$ **глобально недетерминировано**, *если оно не не является глобально детерминированным.*

Множество вычислительных процессов $\Theta^{@}_{\pi}$ **локально детерминировано**, *если мощность* $\Theta^{@}_{\pi}$ *равна единице*: $|\Theta^{@}_{\pi}| = 1$. Иными словами, локальная детерминированность связана с тем, что любое выполнение компьютером @ программы π всегда порождает один и тот же вычислительный процесс α. *Если же* $|\Theta^{@}_{\pi}| > 1$, *то в этом случае* $\Theta^{@}_{\pi}$ **локально недетерминировано**. Могут подумать, что локальная недетерминированность обязательно связана с тем, что выполнение хотя бы одной команды из программы π неоднозначно в отношении наступления следующего события. Однако, как будет показано, это не так.

Очевидно, что глобальная детерминированность зависит от типа реализуемых вычислительных процессов, тогда как локальная детерминированность от типов реализуемых процессов не зависит и может быть определена только на множествах процессов. Ответ на вопрос, какие типы вычислительных процессов из семи возможных реализуемы, и на вопрос, имеются ли локально недетерминированные вычисления, определяется соответствую-

щей теорией вычислимости. Например, теории классической вычислимости, как уже отмечалось, допускают глобальную, но не локальную недетерминированность.

2 Источники недетерминированных вычислений

В самом общем плане источниками недетерминированности могут быть как *внешние*, так и *внутренние* факторы. Разница здесь фундаментальная — такая же, как между учением о том, что мир создан Богом, и концепцией, согласно которой мир самодостаточен и существует сам по себе. В [1] проблема внешнего индетерминизма обсуждалась применительно к классической теории МНР-вычислимости, в которой начальные конфигурации, с которых начинаются вычисления, появляются неизвестно откуда, поистине чудесным образом. Здесь же мы будем рассматривать только внутренние механизмы как детерминированных, так и недетерминированных вычислений. Идею внутренних факторов реализуем при помощи принятия следующей *аксиомы существования*.

АКСИОМА 2. *Каждое событие в вычислительном процессе является результатом выполнения какой-либо команды программы.*

Пусть, например, в языке программирования имеется оператор присваивания $:=$ и индивидная константа c. Аксиома существования запрещает считать программой конструкцию $1.x := y$, поскольку неизвестно, откуда взялось значение y, т.е. отсутствует команда, реализующая событие присваивания значения этой переменной. Но конструкции $1.y := c$ и $1.y := c; 2.x := y$ вполне законны. Нарушает эту аксиому ввод данных с клавиатуры, поскольку события нажатия клавиш чисто внешние и не являются следствиями каких-либо компьютерных команд. Короче говоря, не допускается никакое внешнее вмешательство в вычислительный процесс.

Принятие аксиомы существования позволяет уйти от вопросов о детерминированности или недетерминированности внешних источников событий в вычислительном процессе. Например, если в ходе выполнения программы требуется ввести данные с клавиатуры, то нажимающий на клавиши субъект может решить действовать как детерминированно (скажем, всегда нажи-

мать одну и ту же клавишу), так и недетерминированно (допустим, нажимать клавиши наугад). Понятное дело, сама выполняемая программа эти события никак не контролирует. Было бы неразумно включать в обсуждение вопросов вычислимости проблему свободы воли нажимающего на клавиши субъекта. Но аксиома существования не запрещает наличия команд, имитирующих ввод данных с клавиатуры. В этом случае такие команды должны быть точно синтаксически и семантически описаны, что позволит однозначно решить вопрос об их детерминированности или недетерминированности.

Выделенные в литературе внутренние источники недетерминированной вычислимости, по сути, сводятся к трем разновидностям выходов за пределы детерминизма: вычислимость на отношениях (вместо функций), вероятностные правила вычислений (важный частный случай вычислимости на отношениях) и вычислимость с использованием оракулов. Наличие не взаимнооднозначных функций не рассматривается как проявление недетерминизма. Например, получив в результате выполнения команды $x + y$ число 8, мы остаемся в неведении относительно значений переменных x и y. Однако в реальных вычислениях эти значения будут определены однозначно. Ведь в противном случае выполнить операцию сложения невозможно. Более того, значения переменных x и y сохранятся в памяти компьютера и после выполнения команды сложения, что позволяет, например, повторить эту команду с тем же результатом.

Другое дело, если результат вычислений на определенном входе до выполнения команды остается неопределенным. Любое не функциональное отношение является моделью такой ситуации. Пусть значение переменной x не определено. Тогда команда выполнить предикат $x \leq 8$, рассматриваемая как требование присвоить x значение, меньшее или равное 8, может быть реализована несколькими способами, так что ее результат не однозначен. Это приводит к ситуации $|\Theta_{\pi}^{@}| > 1$ и, таким образом, возникает локальная недетерминированность. Вообще, если команда вычисляет не сингулярное свойство $F(x)$, где $F \subset U$ и $|F| > 1$, выбирая такое x, что $F(x)$, имеем $|\Theta_{\pi}^{@}| > 1$. Аналогичным образом, если вычисляется многоместное отношение F^{n+1} на U, удовлетворяющее отрицанию условия функциональности

$\neg(\forall < x_1, x_2, \ldots, x_n, x_{n+1} > \in F^{n+1})(\forall < y_1, y_2, \ldots, y_n, y_{n+1} > \in F^{n+1})$

$((< x_1, x_2, \ldots, x_n > = < y_1, y_2, \ldots, y_n >) \to (x_{n+1} = y_{n+1}))$,

вновь имеем $|\Theta_\pi^{@}| > 1$, поскольку при $< x_1, x_2, \ldots, x_n > = < y_1, y_2, \ldots, y_n >$ значение $n+1$ компоненты вычисляется (по крайней мере, хотя бы в одном случае) неоднозначно.

В реальных вычислительных устройствах не удается реализовать даже простейший случай бинарной недетерминированности, когда требуется осуществить ни от чего не зависящий выбор одного из двух событий, типа перехода $\alpha \swarrow^{\beta}_{\neg\beta}$. Однако выбор становится реализуемым, если поставить его в зависимость от вероятности наступления одного из двух (или более) событий. Это требует включения в состав компьютера генератора случайных чисел, работающего в зависимости от какого-то случайного физического процесса (например, замер времени между щелчками счетчика Гейгера рядом с образцом изотопа рубидия-85, снятие параметров с нестабильных электрических цепей и т.п.). Формально невероятностная недетерминированность и вероятностная недетерминированность описываются, например, посредством недетерминированной и вероятностной машин Тьюринга. Этот круг вопросов хорошо описан в литературе.

Менее ясна в философском отношении ситуация с использованием в вычислениях оракулов. Применительно к МНР-вычислимости обобщение состоит в переходе к МНРО-вычислимости, где МНРО есть сокращения для «машина с неограниченными регистрами и оракулом». В МНРО наряду со стандартными МНР-командами (обнуления $Z(n)$, прибавления единицы $S(n)$, переадресации $T(m,n)$ и условного перехода $J(m,n,q)$) имеется команда обращения к оракулу $O_\xi(n)$. Встретив эту команду в программе, МНРО текущее содержимое r_n регистра R_n заменяет на значение функции $\xi(r_n)$, где ξ — какая-либо тотальная одноместная функция из N в N. Иными словами, в результате выполнения команды $O_\xi(n)$ осуществляется присваивание $r_n := \xi(r_n)$ (см. [9]). Тем самым стандартная вычислимость обобщается до *ξ-вычислимости* (см. также [15]).

В том случае, если функция ξ является вычислимой, класс вычислимых функций не меняется, т.е. любое МНРО-вычисление

с обращениями к оракулу можно превратить в МНР-вычисление без использования оракула. Но если функция ξ, на которой основывается оракул, оказывается не вычислимой, класс ξ-вычислимых функций окажется шире класса вычислимых в стандартном смысле функций. Например, пусть ξ есть функция, которая определяется следующим образом.

$$\xi(n) = \left\{ \begin{array}{l} 1, \text{ если } n \text{ — гёделев номер истинной формулы} \\ \quad \text{арифметики} \\ 0, \text{ в противном случае} \end{array} \right\}$$

Как известно, множество гёделевых номеров истинных в стандартной модели арифметических формул не только не рекурсивно, но даже не является рекурсивно перечислимым. Однако МНРО-программа π_ξ

1. $O_\xi(1)$

осуществит требуемый пересчет всех арифметических истин в языке формальной арифметики. Достаточно запускать программу π_ξ с начальными конфигурациями $n, 0, 0, 0, \ldots$, где n последовательно принимает значения $0, 1, 2, 3, \ldots$.

Возникает ощущение, что программы, существенно использующие обращения к оракулам, основанным на не вычислимых функциях, демонстрируют недетерминированное поведение. Однако это не так, как показывает следующая теорема.

ТЕОРЕМА 3. *Для всякой тотальной функции ξ из N в N и любой МНРО-программы π_ξ, содержащей команды вида $O_\xi(n)$, имеем $|\Theta_{\pi_\xi}^{MHPO}| = 1$.*

Доказательство. Доказательство основано на том, что функция ξ тотальна, и потому любое применение команды вида $O_\xi(n)$ приведет к однозначному результату на любом шаге вычислений. Q.E.D.

Таким образом, согласно этой теореме, МНРО-вычислимость является локально детерминированной. Но требовать глобальной детерминированности было бы неразумно, поскольку применение команд обращения к оракулу не влияет на возможность появления расходящихся вычислений. Зато теорема заставляет

усомниться в адекватности понятия локальной детерминированности. Когда не вычислимая функция ξ выдает значение, то делает она это непостижимым путем, поистине близким к чуду. А разве чудеса имеют что-то общее с детерминизмом? Вопрос риторический.

Представляется, что здесь мы сталкиваемся с проблемой недетерминизма на двух уровнях: *онтологическом* и *эпистемологическом*. Там и тогда, когда процессы описываются на языке функций, они, с нашей точки зрения, оказываются полностью онтологически детерминированными. Другой вопрос, что многие функциональные процессы (такие, как колебания маятника, преобразование пекаря, образование ячеек Бенара, МНРО-вычислимость и т.д.) невозможно описать в доступных познанию деталях. Это и приводит к эпистемологическому недетерминизму. Малейшая неточность в определении исходных данных (начальных условий) может за короткое время исключить всякую надежду на адекватное предвидение хода процесса. Или, как в случае МНРО-вычислимости, исходные данные точны, но у познающего субъекта в принципе нет общего метода, позволяющего предсказывать результаты МНРО-вычислений. Наши познавательные возможности объективно ограничены — вот в чем суть. Даже если на онтологическом уровне все детерминировано, это еще не означает, что детерминированными будут и познавательные процессы. Разумеется, остается возможность распространить недетерминизм и на онтологический уровень. Достаточно, например, в определении оракула заменить функцию ξ не функциональным отношением или не тотальной функцией.

И все же нам, по-видимому, психологически легче смириться с наличием недетерминированных переходов типа $\alpha \overset{\beta}{\underset{\neg\beta}{\diagup\!\diagdown}}$, чем с оракулом O_ξ, основанным на детерминированной, но не вычислимой функции ξ. Ведь недетерминированные переходы мы можем делать сами в уверенности, что обладаем свободой воли. Но проимитировать поведение не вычислимой и потому весьма сложной функции ξ нам не под силу. Как бы там ни было, мы здесь оставим дальнейшее обсуждение проблем эпистемологического недетерминизма, сосредоточившись на онтологическом уровне.

Существует еще один источник возникновения недетерминизма, доселе ускользавший от внимания исследователей. Речь идет о возможностях исполняющего команды компьютера. Рассмотрим следующую элементарную программу π_G.

1. $GOTO1$

Ясно, что выполнение программы π_G порождает уходящий в бесконечность вычислительный процесс, так что глобально этот процесс не будет детерминированным. Столь же ясным представляется ответ на вопрос, будет ли это выполнение локально детерминированным. Конечно, будет, так как единственно возможной реализацией программы π_G является бесконечная ω-последовательность повторения одного и того же события $S_1, S_2, \ldots, S_n, \ldots$. То есть имеем $S_i = S_j$ при любых i и j, поскольку ничего другого, кроме события безусловного перехода к выполнению первой команды программы, произойти не может. В результате получаем $|\Theta^{@}_{\pi_G}| = 1$. Однако это рассуждение верно лишь при условии, что выбран такой компьютер @, который порождает единственный вычислительный процесс при повторных запусках. Но логически возможен компьютер @, который приводит к неравенству $|\Theta^{@}_{\pi_G}| > 1$! В самом деле, ничто не препятствует предположению, что кроме процесса $S_1, S_2, \ldots, S_n, \ldots$ посредством более мощного компьютера @ может реализоваться не имеющий первого шага выполнения процесс $\ldots, S_{-n}, \ldots, S_{-2}, S_{-1}, S_0, S_1, S_2, \ldots, S_n, \ldots$, упорядоченный по типу $\omega * + \omega$.

Или даже могут реализоваться трансфинитные процессы типа $\omega + \omega * + \omega$, $\omega * + \omega + \omega * + \omega$ и т.п. Что действительно логически невозможно в ходе выполнения π_G, так это завершение вычислительного процесса. Ведь никаких логических оснований, коль скоро рассматриваются не изнашивающиеся и не ломающиеся идеальные компьютеры, для завершения выполнения π_G нет. Логически невозможны также процессы, содержащие особые нарушающие дискретность события, например, упорядочения типа $\omega + \omega$, как об этом уже говорилось в предыдущем параграфе.

Могут сказать, что невозможно представить, как процесс, порождающий шаг за шагом неограниченный ряд ω, может оказаться в области событий из $\omega *$. Ведь ряд вида $S_0, S_1, S_2, \ldots, S_n, \ldots, S_{-n}, \ldots, S_{-2}, S_{-1}$, упорядоченный по типу $\omega + \omega *$, обладает

тем свойством, что любые два события S_j и S_{-j} разделяет бесконечное число шагов! Согласен, *представить* это нельзя. Но с *абстрактной* точки зрения ничего логически невозможного здесь нет. Просто не надо думать, что каждый шаг вычислений требует затрат какого-то отличного от нуля времени. А если допустить, что время выполнения каждой отдельной команды на сверхмощном компьютере равно нулю?

3 Причинность, вычислимость и производительность

В ходе философского осмысления человеческого опыта и результатов наук накапливается все больше проблем, обсуждение которых требует привлечения альтернативных теорий вычислимости. Возьмем старую философскую проблему причинности. Победоносное шествие юмовского подхода к причинности как функции, при котором проблема производительной причинности (в смысле причина *производит* следствие) объявляется псевдопроблемой, обусловлено отсутствием надлежащего концептуального аппарата. В подходящей альтернативной теории вычислимости понятию «производительной причинности» можно придать точный смысл.

В свое время Платон определил причину как «то, что творит». Представление о том, что причина творит, порождает или производит следствие, господствовало в течение веков до тех пор, пока Д. Юм не обрушился на него с уничтожающей критикой. Взгляды Юма на причинность разделялись философами и учеными позитивистской направленности, развившими аргументацию английского философа. Согласно неопозитивистам, в математических формулировках научных теорий фиксируется неизменная последовательность состояний природных явлений во времени, и это все, что остается от причинной связи. Подчеркивается, что в математическом аппарате научных теорий нет ничего, даже отдаленно напоминающего производительность. Более того, само понятие причинности иногда объявляется не только излишним, но и вредным для науки (Б. Рассел). Точка зрения Рассела состояла в том, что понятие причинности должно уступить место понятию функции. Однако, как примирительно заметил Г.Х. фон Вригт, хотя «понятие причины не играет важной роли

в ведущих теоретических науках и в этих науках вполне можно использовать функциональную терминологию вместо каузальной, остается фактом, что каузальное мышление как таковое не изгоняется из науки подобно злому духу, а следовательно, философские проблемы причинности остаются центральными в философии науки»([7, с. 73-74]).

Для М.Бунге (см. [5]) неопозитивистская трактовка математического инструментария науки неприемлема, так как он под причинностью имеет в виду не просто следование явлений во времени, а производство одних явлений другими. Бунге справедливо указывает, что без производительного аспекта понятие причинности теряет смысл. Но в используемых в науке математических формулах нет и намека на производительность. Кстати говоря, именно последнее обстоятельство служит основанием позитивистской позиции в большей степени, чем что-либо другое. Ведь если бы удалось в математическом формализме науки найти адекватное выражение идеи производительной причинности, то позитивизм столкнулся бы с серьезными проблемами. Но чего нет, того нет: логика и математика (по крайней мере, до самого последнего времени) не имели средств для описания того, как одно явление производит или порождает другое.

Этому факту имеется объяснение. Стандартное научное описание природных явлений основывается на математическом понятии функции. Функциональный язык способен выразить идею изменений в форме «Всякий раз, когда имеет место явление A, имеет место (тут же или с запаздыванием во времени) явление B». Но не в форме «Явление A производит явление B», т.е. не в форме описания процесса производства или порождения одним явлением другого. Выражения типа «функция F, будучи примененной к аргументу x, дает в результате y» могут при поверхностном подходе навести на мысль, что функция F каким-то образом порождает y из x. На самом деле ничего подобного нет. Мы, конечно, можем сказать, что $F(x) = y$ и тогда, когда, как мы считаем, явление, обозначенное через x, действительно порождает явление, обозначенное через y. Но в самом «механизме» функционального описания процесс порождения никак не представлен. Можно утверждать лишь то, что представлены начальный (аргумент функции) и конечный (значение функции)

результаты некоторого процесса. Но можно этого и не утверждать, отбросив процессуальную терминологию как излишнюю, коль скоро мы удовлетворяемся функциональными описаниями. Лишь в последнее время всерьез начинают рассматривать возможность описания не функции, связывающей изучаемые явления, а описание алгоритма, лежащего в основе связи явлений. Программный, алгоритмический язык богаче функционального, поскольку для каждой вычислимой функции F такой, что $F(x) = y$, имеется бесконечно много способов получить (произвести!) y из x, выбрав соответствующую компьютерную программу. Иными словами, для каждой вычислимой функции имеется бесконечный класс программ, которые ее вычисляют. Функция будет одной и той же, а процесс порождения из аргумента соответствующего значения может существенно отличаться по важным параметрам (например, скорости выполнения) для разных программ ее вычисления.

Приведенные (конечно, слишком краткие) доводы показывают, что классический функциональный математический аппарат науки просто не обладает достаточной выразительной силой, чтобы описать в адекватной форме идею производительной причинности. Конечно, если в реальной науке нет производящей причинности, то незачем вводить ее извне. Если же мы настаиваем на порождающей причинности по философским соображениям, от нас имеют право потребовать объяснить в точных терминах, что мы под этим понимаем. Впрочем, кажется, что есть шансы на введение производящей причинности в науку, что нами связывается с утверждением представления о наличии у реальности вычислительного (производящего!) аспекта. В настоящее время в науке все более широкое распространение получает идея о том, что реальность можно описывать средствами теории вычислений. Компьютерные модели реальности вызывают возрастающий интерес, особенно в биологии и физике.

Назовем утверждение о событии Q *фактуальным*, если Q может быть как истинным, так и ложным в зависимости от наступления или не наступления данного события. Пусть фактуальное утверждение P является истинным до начала выполнения программы π, и программа π начинает выполняться компьютером @. Запишем это высказывание как (P, π) и назовем P *предусло-*

вием программы π. Пусть, далее, выполнение программы π обязательно заканчивается (обозначим это как $\pi\downarrow$), и фактуальное утверждение S о каком-то событии истинно после выполнения программы π. В этом случае пишем (π, S) и назовем S *постусловием* программы π. Точнее, надо было бы писать $(\pi\downarrow, S)$, однако в этом параграфе нас будут интересовать только такие программы π, для которых верно $\pi\downarrow$, ибо в противном случае понятие постусловия делается бессмысленным. Аналогичным образом, если процесс выполнения π не имеет первого шага, т.е. если $\uparrow\pi$, бессмысленным делается понятие предусловия. Так что стрелки $\uparrow$ и $\downarrow$ можно не использовать. Если исключить также возможность авоста, то сказанное означает, что *причинные связи глобально детерминированы*.

Возможно, что (P, π) и затем (π, S) при каком-то выполнении программы π, однако при локально недетерминированном выполнении программы π компьютером @ в следующий раз при наличии (P, π) мы можем получить $(\pi, \neg S)$. Назовем программу π *корректной* относительно высказываний P и S (P и S берутся в указанном порядке), если истинность (P, π) влечет истинность (π, S) при любом варианте выполнения программы π компьютером @. Утверждение о корректности программы π относительно предусловия P и постусловия S будем записывать в виде $P\{\pi\}S$.

Введем отношение эквивалентности на множестве программ. Программы π и τ *эквивалентны* ($\pi \equiv \tau$), если и только если для любых высказываний о событиях P и S имеет место $(P\{\pi\}S \leftrightarrow P\{\tau\}S)$. Иными словами, программы эквивалентны, если они корректны для одних и тех же пар пред- и постусловий. В том, что приведенное определение действительно задает отношение эквивалентности, легко убедиться, используя свойства логической связки $\leftrightarrow$. Во многих случаях достаточно рассматривать программы с точностью до эквивалентных.

Некоторые программы могут выполняться последовательно, одна за другой. Такое последовательное выполнение иногда возможно даже в тех случаях, когда первая программа порождает процесс, не имеющий последнего шага, а вторая — первого шага выполнения, как об этом говорилось в предыдущем параграфе. Опуская здесь такие случаи, ограничимся ситуацией, когда обе программы порождают только процессы, имеющие первый

и последний шаги выполнения. При этом допущении мы вправе писать $(\pi;\tau)$ (читается: «выполнить π, затем выполнить τ»), считая, что $(\pi;\tau)$ также является программой.

Для анализа каузальности в первую очередь важны имеющие начало и когда-либо завершающиеся процессы. Действительно, если мы хотим вести речь о причинах и их следствиях, то должны быть события, имевшие место до начала программы, и события, наступившие после ее завершения. Однако мы отказываемся от традиционной точки зрения, рассматривавшей в качестве причин и следствий только события. Наша позиция, коротко говоря, состоит в том, что события сами по себе не могут быть причинами других событий или их следствиями. Если Д. Юм и те, кто приняли его теорию причинности (точнее, отсутствия причинности), настаивали именно на этом, то с ними можно согласиться. Но если затем принимался (явно или неявно) тезис, согласно которому проблема причинности сводится к отношениям между событиями и только, то этот тезис неверен.

Появление или исчезновение событий — результат действия происходящих в природе и обществе процессов, причем процессы производят не только события, но и другие процессы. Это с одной стороны. С другой стороны, события оказывают влияние на происходящие процессы, вплоть до блокирования одних процессов и инициирования других. Таким образом, необходимо исследовать отношения событий и процессов к событиям и процессам. Одним из ключевых отношений здесь оказывается отношение причинной связи. В рамках сформулированной идеи возможны различные формальные подходы к каузальности, что должно стать предметом дальнейшего исследования. Здесь мы ограничимся (используя ранее введенные понятия и соглашения) кратким изложением исходных шагов построения одной конкретной теории причинности, которая, на наш взгляд, заслуживает внимания.

Будем говорить, что (P,π) *является* **причиной** (τ,S) и использовать запись $(P,\pi)\Longrightarrow(\tau,S)$, *если существуют фактуальные утверждения* Q *и* R *такие, что* $Q\leftrightarrow R$, $P\{\pi\}Q$ *и* $R\{\tau\}S$.

С содержательной точки зрения смысл данного определения в том, что событие, зафиксированное в предложении P, иници-

ирует процесс выполнения π, который завершается событием, фиксируемом предложением Q. Чтобы не громоздить обозначения, мы будем обозначать события теми же символами, что и предложения, их фиксирующие. Поскольку Q и R эквивалентны, они указывают (по принимаемому допущению) на близкие по сути события, так что событие Q, завершившее процесс выполнения π, приводит к наступлению события R, запускающего процесс выполнения программы τ, который, в свою очередь, завершается событием S. Поэтому, чтобы получить процесс выполнения τ и событие S, достаточно иметь событие P и процесс выполнения π. Одного события P или одной программы π по отдельности, вообще говоря, недостаточно ни для запуска выполнения программы τ, ни для получения S. Можно (в принципе) проследить шаг за шагом, каким образом P и π производят τ и S, существуют ли более эффективные в том или ином смысле способы порождения τ и S, какие побочные следствия возникают в ходе протекания рассматриваемых процессов и тому подобное. При функциональном подходе все эти и подобные вопросы, по меньшей мере, неуместны или даже бессмысленны.

При описании причинных связей удобно иметь в своем распоряжении процесс, который ничего не меняет (аналог 0 в алгебре). Постулируется, что имеется элементарный процесс $skip$, такой, что для любого предложения о событиях P имеет место $P\{skip\}P$.

ТЕОРЕМА 4. *Для любой программы π, для которой имеет место $\downarrow \pi \downarrow$, верны следующие эквивалентности: $\pi \equiv (skip; \pi)$ и $\pi \equiv (\pi; skip)$.*

Доказательство очевидное. Отметим, однако, что пункты о первом и последнем шаге существенны. Если процесс выполнения π не имеет начала, то конструкция $(skip; \pi)$ не является вычислительным процессом, поскольку не является линейной дискретной последовательностью шагов (нарушается условие 4). Аналогичным образом, $(\pi; skip)$ также не процесс в случае, если выполнение π не имеет последнего шага (нарушается условие 5).

ТЕОРЕМА 5. *Если $P\{\pi\}S$, то $(P, skip) \Longrightarrow (\pi, S)$ и $(P, \pi) \Longrightarrow (skip, S)$.*

Доказательство. Допустим, что верно $P\{\pi\}S$. Поскольку $P\{skip\}P$ и $P \leftrightarrow P$, получаем $(P, skip) \Longrightarrow (\pi, S)$. Аналогичным образом, $P\{\pi\}S$ и $S\{skip\}S$ дают $(P, \pi) \Longrightarrow (skip, S)$. Q.E.D.

ТЕОРЕМА 6. *Отношение* $\Longrightarrow$ *не является ни рефлексивным, ни симметричным, ни транзитивным.*

Доказательство. Действительно, запись вида $A \Longrightarrow A$ не верна синтаксически. Точно так же из двух форм $A \Longrightarrow B$ и $B \Longrightarrow A$, по крайней мере, одна нарушает синтаксические правила записи о причинных связях. То же самое верно в отношении записей вида $A \Longrightarrow B, B \Longrightarrow C \ldots$. Q.E.D.

Если в отношении свойств рефлексивности и симметричности последний факт, по-видимому, не будет вызывать возражений, то отсутствие свойства транзитивности причинных связей может отвергаться по интуитивным соображениям. Однако интуиция здесь — плохой советчик. Запись вида $\ldots \Longrightarrow B$ указывает на завершение некоторого процесса, а запись вида $B \Longrightarrow \ldots$ — на начало некоторого процесса. Поэтому одно и то же B не может фигурировать и в качестве следствия, и в качестве причины. Причинно-следственные цепочки строятся иначе. Прежде чем вернуться к обсуждаемому вопросу, сделаем одно отступление.

ТЕОРЕМА 7. *Если* $(P, \pi) \Longrightarrow (\tau, S)$, *то* $P\{\pi; \tau\}S$.

Доказательство. По определению причинности, существуют предложения Q и R, такие, что $Q \leftrightarrow R$, $P\{\pi\}Q$ и $R\{\tau\}S$. Из этого следует, что событие P начинает процесс выполнения π, который завершается событием Q, и, в силу эквивалентности Q и R, также событием R, которое инициирует процесс выполнения τ, завершающийся событием S, что и требовалось. Q.E.D.

Следующий факт является своего рода аналогом транзитивности.

ТЕОРЕМА 8. *Пусть* $(P, \pi) \Longrightarrow (\sigma, Q)$ *и* $(Q, \varsigma) \Longrightarrow (\tau, S)$. *Тогда* $(P, \pi; \sigma) \Longrightarrow (\varsigma; \tau, S)$.

Доказательство. По только что доказанному имеем $P\{\pi;\sigma\}Q$ и $Q\{\varsigma;\tau\}S$. Применяя определение причинности, получаем $(P,\pi;\sigma) \Longrightarrow (\varsigma;\tau,S)$. Q.E.D.

При этом сохраняется возможность традиционного неточного словоупотребления: коль скоро речь идет только о событиях, в случае, если P причина Q и Q причина S, будет иметь место P причина S. Однако такой способ речи, упускающий из виду процессуальную сторону причинной связи, возвращает нас к функциональному взгляду на каузальность, при котором теряется специфическая особенность причинности — производительность.

ТЕОРЕМА 9. *Для любых* @, P, S , π *и* τ, *если* $(P,\pi) \Longrightarrow (\tau,S)$, *то множества* $\Theta^{@}_{\pi}$ *и* $\Theta^{@}_{\tau}$ *глобально детерминированы, но для некоторых* @, P, S , π *и* τ *при* $(P,\pi) \Longrightarrow (\tau,S)$ *множество* $\Theta^{@}_{\pi}$ *или множество* $\Theta^{@}_{\tau}$ *локально недетерминировано.*

Доказательство. Глобальная детерминированность множеств вычислительных процессов $\Theta^{@}_{\pi}$ и $\Theta^{@}_{\tau}$ является прямым следствием определений. Что касается возможной локальной недетерминированности, то она может быть вызвана недетерминированой командой одной из программ, как показывает следующий пример.

Пусть P есть сокращение для $x < 1$, π — присваивание $x := 1$, τ — присваивание $x := 2\ OR\ x := 3$, выполняемое компьютером @ недетерминированым образом, и S есть $x > 1$. В качестве как Q, так и R возьмем $x = 1$. Отсюда имеем $Q \leftrightarrow R$, $P\{\pi\}Q$ и $R\{\tau\}S$, что дает $(P,\pi) \Longrightarrow (\tau,S)$. Однако, очевидном образом, имеем $|\Theta^{@}_{\tau}| > 1$. Q.E.D.

Этот факт оправдывает утвердившийся в философии науки термин «вероятностная причинность», позволяя придать ему точный процедурный смысл. В рассматриваемом примере достаточно приписать двум возможным результатам выполнения команды из τ, скажем, равные вероятности. Да и на обыденном уровне люди рассуждают о причинах и следствиях в условиях неоднозначной связи между ними. Так, родители естественным образом выступают в качестве порождающей причины своих детей, но пол будущего ребенка заранее не определен.

Литература

[1] *Анисов А.М.* Классическая вычислимость и признаки индетерминизма // Логические исследования. Вып. 14. М., 2007.

[2] *Анисов А.М.* Темпоральный универсум и его познание. М., 2000.

[3] *Брауэр В.* Введение в теорию конечных автоматов. М., 1987.

[4] *Булос Дж., Джеффри Р.* Вычислимость и логика. М., 1994.

[5] *Бунге М.* Причинность. Место принципа причинности в современной науке. М., 1962.

[6] *Верещагин Н.К., Шень А.Х.* Лекции по математической логике и теории алгоритмов. Ч. 3. Вычислимые функции. М., 2002.

[7] *Вригт Г.Х. фон.* Логико-философские исследования. М.,1986.

[8] *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.

[9] *Катленд Н.* Вычислимость. Введение в теорию рекурсивных функций. М., 1983.

[10] *Китаев А.Ю., Шень А.Х., Вялый М.Н.* Классические и квантовые вычисления. М., 1999.

[11] *Кузюрин Н.Н., Фомин С.А.* Эффективные алгоритмы и сложность вычислений. Версия от 17.06.2008. 348 стр. http://discopal.ispras.ru/ru.book-advanced-algorithms.htm.

[12] *Маслов С.Ю.* Теория дедуктивных систем и ее применения. М., 1986.

[13] *Хармут Х.* Применение методов теории информации в физике. М., 1989.

[14] *Хоар Ч.* Взаимодействующие последовательные процессы. М., 1989.

[15] *Шенфилд Дж.* Степени неразрешимости. М., 1977.

[16] *Шор Р.* Теория α-рекурсии // Справочная книга по математической логике: В 4 частях. Ч. III. Теория рекурсии. М., 1982.

[17] *Якобс К.* Машины Тьюринга и случайные 0-1-последовательности // Машины Тьюринга и рекурсивные функции. М., 1972.