НОРМАТИВНЫЙ КОНСПЕКТ ПО ДИСЦИПЛИНЕ «ИНТЕЛЛЕКТУАЛЬНЫЕ СИСТЕМЫ В ГУМАНИТАРНОЙ СФЕРЕ»

Часть 2
«АЛГОРИТМИЧЕСКАЯ МОДЕЛЬ ПРЕДСТАВЛЕНИЯ ЗНАНИЙ. МАШИНЫ ТЬЮРИНГА»

Введение. Неформально (то есть математически неточно) под *алгоритмом* понимается предписание, определяющее вычислительный процесс, ведущий от варьируемых исходных данных к искомому результату. В алгоритмической модели представления знаний под *знанием* понимается *конкретное точное предписание*, а *оперирование знаниями* означает *применение этого точного предписания* к исходным данным с целью получения вполне конкретного искомого результата.

Для алгоритмов характерны три важнейших свойства: *определенность* – предписание должно быть точным, не оставляющим места произволу; *массовость* – алгоритм должен давать возможность исходить из варьируемых в известных пределах исходных данных; *результативность* – направленность алгоритма на получение некоторого искомого результата, в конце концов и получаемого при надлежащих исходных данных.

Можно пользоваться неформальным определением понятия «алгоритм», пока термин «алгоритм» встречается в лишь в положительных высказываниях типа «для решения таких-то задач имеется алгоритм, и вот в чем он состоит». Однако, могут существовать задачи, относительно которых у нас складывается впечатление, что они *алгоритмически неразрешимы* (то есть алгоритма для их решения не существует). Никакие отрицательные результаты, никакие теоремы невозможности алгоритмов не могли бы быть *доказаны* с использованием неформального понятия алгоритма.

Свой вариант точного (*формального*, математического) определения понятия «алгоритм» предложил выдающийся британский математик Алан Тьюринг в 1936 году в статье «On computable numbers, with an application to the entscheidungsproblem» («О вычислимых числах с приложением к проблеме разрешимости»). Происхождение названия статьи и научного интереса Тьюринга таково: еще в 1900 году на II Международном Конгрессе математиков в Париже выдающийся немецкий «король математиков» Давид Гильберт прочитал доклад «Математические проблемы», в котором предложил 23 кардинальные (по его мнению) нерешенные (на тот момент времени) математические проблемы. Во введении к этому докладу (и при формулировке 10-й проблемы) Гильберт высказал мнение, состоящее в том, что если некоторая массовая задача имеет решение в каждом своем частном случае (то есть является вообще *разрешимой*), то она является и *алгоритмически разрешимой*, то есть обязательно может быть найден алгоритм, решающий эту задачу в общем виде.

Невзирая на авторитет Гильберта, многие математики усомнились в справедливости высказанной им гипотезы. Они считали, что существуют такие массовые задачи, которые действительно имеют решение в каждом своем частном случае (то есть вообще разрешимы), но являются *алгоритмически неразрешимыми*, то есть для них не существует алгоритма, решающего задачу в общем виде.

Так возникла проблема, получившая название *проблемы разрешимости* (по-немецки «entscheidungsproblem»):

- *либо* ***доказать****, что каждая* ***вообще разрешимая*** *массовая задача является также и* ***алгоритмически разрешимой****;*

- *либо привести пример (хотя бы один) такой массовой задачи, которая является* ***алгоритмически неразрешимой****, невзирая на то, что она* ***вообще разрешима***.

Алан Тьюринг вознамерился привести пример, требуемый во второй части данной формулировки: сформулировать некоторую массовую задачу; доказать, что она вообще разрешима; доказать, что не существует решающего ее алгоритма. Как мы уже отмечали выше, достигнуть этой цели Тьюринг не смог бы, если бы не предложил свою формализацию понятия «алгоритм». Сейчас эта формализация называется *машиной Тьюринга*. С тьюринговой точки зрения «сформулировать алгоритм» означает «построить машину Тьюринга», «доказать существование алгоритма» означает «доказать существование машины Тьюринга», «доказать алгоритмическую неразрешимость массовой задачи» означает «доказать, что не существует машины Тьюринга, решающей эту задачу».

Определение машины Тьюринга. Машина Тьюринга есть математическая (воображаемая) машина, а не машина физическая. Тьюринг предпринял попытку смоделировать действия математика (или другого человека), осуществляющего некую интеллектуальную созидательную деятельность.

Такой человек, находясь в определенном «умонастроении» («состоянии»), просматривает некоторый текст. Затем он вносит в этот текст какие-то изменения, проникается новым «умонастроением» и переходит к просмотру последующих записей. Машина Тьюринга действует примерно так же. Ее удобно представлять в виде автоматически работающего устройства. На каждом такте своей работы устройство, находясь в некотором *внутреннем состоянии*, обозревает содержимое одной ячейки протягиваемой через устройство ленты и делает шаг, заключающийся в том, что устройство переходит в новое внутреннее состояние (или остается в прежнем внутреннем состоянии), изменяет (или оставляет без изменения) содержимое обозреваемой ячейки и переходит к обозрению следующей ячейки – справа или слева (или «решает остаться на месте»).

Каждый шаг осуществляется на основании предписанной *команды*. Совокупность всех команд представляет собой *функциональную схему* или *программу* машины Тьюринга.

Опишем теперь машину Тьюринга более строго. Машина располагает конечным числом знаков (символов, букв), образующих так называемый *внешний алфавит* $A = \{a_0, a_1, ..., a_n\}$. В каждую ячейку обозреваемой ленты на каждом такте

работы может (и должен) быть записан один и только один символ из алфавита A. Удобно считать, что среди букв внешнего алфавита A имеется «пустая буква» («пробел»), и именно она записана в каждую пустую ячейку ленты. Условимся, что «пустой буквой» или символом пустой ячейки является буква a_0. Лента предполагается неограниченной в обе стороны (и влево, и вправо), но в каждый момент времени на ней записано конечное число непустых букв.

В каждый момент времени машина способна находиться в одном внутреннем состоянии из конечного числа *внутренних состояний*, совокупность которых $Q = \{q_0, q_1, ..., q_m\}$. Среди внутренних состояний выделяются два – *начальное* q_1 и *заключительное* (или *состояние остановки*) q_0. Находясь в состоянии q_1, машина начинает работать. Попав в состояние q_0, машина останавливается.

Работа машины определяется *программой* (*функциональной схемой*). Программа состоит из *команд*. Каждая команда представляет собой выражение одного из следующих трех видов:

- $q_i a_j \rightarrow q_k a_l S$ – если, находясь во внутреннем состоянии q_i, машина обозревает ячейку, в которой записан символ a_j, то она должна в эту ячейку записать символ a_l (быть может, и совпадающий с символом a_j) и изменить свое внутреннее состояние на q_k (быть может, и совпадающее с состоянием q_i); смещаться и просматривать соседнюю ячейку не следует (следует «остаться на месте»); в дальнейшем, если это не вызывает недоразумений, символ S в выражениях этого вида будем опускать;

- $q_i a_j \rightarrow q_k a_l R$ – если, находясь во внутреннем состоянии q_i, машина обозревает ячейку, в которой записан символ a_j, то она должна в эту ячейку записать символ a_l (быть может, и совпадающий с символом a_j) и изменить свое внутреннее состояние на q_k (быть может, и совпадающее с состоянием q_i); затем следует перейти к рассмотрению соседней ячейки справа (следует «сместиться на одну клетку вправо»);

- $q_i a_j \rightarrow q_k a_l L$ – если, находясь во внутреннем состоянии q_i, машина обозревает ячейку, в которой записан символ a_j, то она должна в эту ячейку записать символ a_l (быть может, и совпадающий с символом a_j) и изменить свое внутреннее состояние на q_k (быть может, и совпадающее с состоянием q_i); затем следует перейти к рассмотрению соседней ячейки слева (следует «сместиться на одну клетку влево»);

Находясь на каком-либо такте работы в *незаключительном внутреннем состоянии* (то есть во внутреннем состоянии, отличном от q_0), машина совершает шаг, который полностью определяется ее текущим внутренним состоянием q_i и символом a_j, воспринимаемым ею в данный момент на ленте. На следующем такте работы машина снова делает шаг, регламентированный подходящей командой и так далее.

Поскольку «поведение» машины полностью определяется ее внутренним состоянием в данный момент времени и содержимым обозреваемой в этот момент времени ячейки, то для каждого внутреннего состояния программа машины должна содержать одну и только одну команду, начинающуюся символом этого внутреннего состояния и некоторым конкретным символом внешнего алфавита.

Под *конфигурацией* будем понимать изображение ленты машины с информацией, сложившейся на ней к началу некоторого шага (или слово во внешнем алфавите, записанное на ленту к началу этого шага), с указанием того, какая ячейка обозревается в этот момент времени и в каком внутреннем состоянии находится машина. Имеют смысл лишь *конечные конфигурации*, то есть такие, в которых все ячейки ленты, за исключением, быть может, конечного их числа, пусты. Конфигурация называется *заключительной конфигурацией*, если внутреннее состояние, в котором при этом находится машина, является заключительным (q_0).

Если выбрать какую-либо *незаключительную конфигурацию* машины Тьюринга в качестве исходной, то работа машины будет состоять в том, чтобы последовательно (шаг за шагом) преобразовывать исходную конфигурацию в соответствии с программой машины до тех пор, пока не будет достигнута заключительная конфигурация. После этого работа машины Тьюринга считается закончившейся, а результатом работы считается достигнутая заключительная конфигурация.

Во многих задачах на конструирование машин Тьюринга говорят, что непустое слово во внешнем алфавите воспринимается машиной *в стандартном положении*, если оно записано в последовательных ячейках ленты, все другие ячейки пусты, и машина обозревает крайнюю слева ячейку из тех, в которых записано непустое слово.

Стандартное положение называется *начальным*, если машина, воспринимающая слово в этом стандартном положении, находится в начальном внутреннем состоянии (то есть q_1). Стандартное положение называется *заключительным*, если машина, воспринимающая слово в этом стандартном положении, находится в заключительном внутреннем состоянии (то есть q_0).

Будем говорить, что *входное* слово α перерабатывается машиной в *выходное* слово β, если от слова α, воспринимаемого в начальном стандартном положении, машина после выполнения конечного числа команд приходит к слову β, воспринимаемому в заключительном положении.

Тезис Тьюринга. Машина Тьюринга является формализацией понятия «алгоритм». Одно из свойств алгоритма заключается в том, что алгоритм представляет собой единый способ, позволяющий для каждого частного случая массовой задачи за конечное число шагов найти решение. Каждый частный случай массовой задачи можно выразить (закодировать) некоторым словом некоторого алфавита, а решение – каким-то другим словом того же алфавита. В результате получим функцию, заданную на некотором подмножестве множества всех слов выбранного алфавита и принимающую значения в множестве всех слов того же алфавита.

Решить какую-либо задачу в частном случае – значит найти значение этой функции на слове, кодирующем данный частный случай. А иметь алгоритм для решения массовой задачи – значит иметь единый способ, позволяющий за конечное число шагов «вычислять» значения построенной функции для любых значений аргумента из ее области определения. Таким образом, проблема поиска алгоритма

есть проблема вычисления значений функции, заданной для слов в некотором алфавите.

Вычислять значения функции значит вычислять их с помощью подходящей машины Тьюринга. Каждая ли функция, для вычисления значений которой существует какой-нибудь алгоритм, окажется вычислимой посредством некоторой машины Тьюринга? Тьюринг высказал гипотезу, называемую *основной гипотезой теории алгоритмов* или *тезисом Тьюринга*: *для нахождения значений функции, заданной в некотором алфавите, тогда и только тогда существует какой-нибудь* ***алгоритм****, когда она может вычисляться на подходящей* ***машине Тьюринга***.

Это означает, что строго математическое понятие вычислимой (по Тьюрингу) функции является по существу *моделью* взятого из опыта (и математически неточного) понятия алгоритма. Данный тезис есть не что иное, как *аксиома*, постулат, выдвигаемый нами. Данный тезис в принципе не может быть доказан методами математики, потому что он не имеет внутриматематического характера (одна сторона в тезисе – понятие алгоритма – не является точным математическим понятием). Он выдвинут исходя из опыта, и именно опыт подтверждает его состоятельность.

Но теперь имеется принципиальная возможность строго математически сформулировать проблему разрешимости: *может ли быть так, чтобы некоторая функция, устанавливающая соответствие между словами некоторого алфавита существовала, но невозможно было бы построить машину Тьюринга, корректно перерабатывающую каждое возможное значение аргумента этой функции в ее верное значение?*

Тьюрингу удалось построить пример такой функции.

Пример вообще вычислимой функции, невычислимой по Тьюрингу. Пусть областью определения функции $\psi(\alpha)$ является множество непустых слов в алфавите $\{1\}$. Вот представители области определения: $1^1=1$, $1^2=11$, $1^3=111$, $1^4=1111$ и другие им подобные. Определена функция $\psi(\alpha)$ следующим образом.

Множество всех мыслимых машин Тьюринга счетно: каждой машине Тьюринга можно присвоить номер (то есть в принципе можно выполнить *нумерацию* всех машин Тьюринга). Пусть в такой нумерации машинам Тьюринга даны имена MT_1, MT_2, MT_3 и так далее. Символом β_n ($n = 1, 2, 3, \ldots$) обозначим слово, в которое машина Тьюринга MT_n перерабатывает слово 1^n, если, воспринимая это слово 1^n в качестве входных данных, она завершает работу после конечного числа тактов: $\beta_n = MT_n(1^n)$. Если же машина Тьюринга MT_n на слове 1^n зацикливается (то есть не завершает работу после конечного числа тактов), то будем считать, что β_n не определено (не существует). Определим теперь функцию $\psi(\alpha)$ следующим образом: если слово β_n существует, то $\psi(\alpha) = \psi(1^n) = \beta_n 1$. Иными словами, если слово β_n существует, то функция $\psi(\alpha)$ слову вида 1^n ставит в соответствие слово β_n, к которому в конце приписана единица. Если же слово β_n не существует, то пусть функция $\psi(\alpha)$ слову вида 1^n ставит в соответствие однобуквенное слово 1. Ясно, что функция $\psi(\alpha)$ тем самым однозначно определена (то есть она существует, она является вычислимой). Но машины Тьюринга, кор-

ректно перерабатывающей каждое возможное значение аргумента функции $\psi(\alpha)$ в ее верное значение, не существует.

Действительно, если бы такая машина Тьюринга существовала, то в названной выше полной нумерации всех машин Тьюринга ей были бы присвоены некоторый номер k и имя MT_k. Машина Тьюринга MT_k не должна зацикливаться на слове 1^k, ведь значение $\psi(1^k)$ однозначно определено, а машина MT_k, если она существует, должна вычислять значения функции $\psi(\alpha)$ для всех допустимых значений ее аргумента. В соответствии со смыслом символа β_k можно утверждать, что $MT_k(1^k) = \beta_k$. Но в соответствии с тем фактом, что машина Тьюринга MT_k вычисляет значения функции $\psi(\alpha)$, следует утверждать, что $MT_k(1^k) = \beta_k 1$. Слова β_k и $\beta_k 1$ существенно различны, они совместно не могут рассматриваться как одно и то же выходное слово, в которое машина Тьюринга MT_k перерабатывает входное слово 1^k. Полученное противоречие можно устранить, только лишь признав, что рассмотренная нами машина Тьюринга не существует (а значит номер k и имя MT_k в использованной нами нумерации принадлежат какой-то другой, действительно существующей машине Тьюринга, и она «занимается» чем-то другим, а не вычислением значений функции $\psi(\alpha)$, определенной нами).

Итак, задача вычисления значений определенной нами функции $\psi(\alpha)$, хотя и вообще разрешима, но неразрешима с помощью машины Тьюринга. В соответствии с тезисом Тьюринга, тогда и вовсе не существует никакого *алгоритма*, решающего эту задачу как массовую проблему. Такую проблему следует признать *алгоритмически неразрешимой* проблемой.

Можно сказать, что построенный выше пример дает решение поставленной Гильбертом проблемы разрешимости. Проблема разрешимости требует найти ответ на вопрос: существуют ли вообще разрешимые проблемы, неразрешимые алгоритмически? Гильберт считал, что нет. Его гипотезы опровергнута. Вообще разрешимые проблемы, неразрешимые алгоритмически, существуют.

Алгоритмическая неразрешимость проблемы самоприменимости. Внешний алфавит машины Тьюринга содержит не менее двух символов. Одним из них является пустой символ (пробел, символ a_0). Второй для общности рассмотрения будем обозначать a_1. Возможно, во внешнем алфавите машины имеются и другие символы. Кроме того, во внутреннем алфавите машины имеются символы для обозначения внутренних состояний ($q_1, q_2, \ldots, q_n, q_0$) и символы для обозначения указаний о сдвиге головки (L, S, R). Можно говорить, что имеется еще один символ (символ →), играющий вспомогательную роль при записи команд функциональной схемы машины. Общее число названных символов конечно, поэтому вполне можно построить разделимый код, в котором все эти символы кодируются словами (кодонами) в алфавите, содержащем всего два символа a_0 и a_1. Получается, что функциональную схему (программу) машины Тьюринга можно записать в закодированном (и однозначно корректно декодируемом) виде как слово в алфавите $\{a_0, a_1\}$. Будем это слово называть *записью* программы машины.

Предположим, что на ленте машины Тьюринга дана запись ее собственной программы. Если машина Тьюринга применима к такому слову (то есть, начав

его обработку, завершит ее через конечное число тактов), то будем называть ее *самоприменимой*, в противном случае – *несамоприменимой*. Возникает массовая задача – *проблема распознавания самоприменимых машин Тьюринга*. Задача состоит в следующем: по заданной функциональной схеме (программе) машины Тьюринга установить, к какому классу относится машина (к классу самоприменимых машин или к классу несамоприменимых машин).

Ясно, что в каждом частном случае проблема распознавания самоприменимых машин Тьюринга разрешима (то есть вообще разрешима). Действительно, если дана функциональная схема (программа) конкретной машины Тьюринга, то можно построить запись ее программы и представить ее на ленте в качестве входных данных. Затем непременно случится одно из двух: либо машина, начав обработку входных данных, завершит ее за конечное число тактов и будет признана самоприменимой, либо машина зациклится и будет признана несамоприменимой. Иными словами, функция, ставящая в соответствие каждой записи программы машины Тьюринга слово «да» (для самоприменимых машин) или «нет» (для несамоприменимых машин), несомненно, существует и вычислима. Однако, машины Тьюринга, вычисляющей эту функцию, не существует. Докажем это.

Допустим противное, то есть пусть существует такая машина Тьюринга Θ, которая запись $D(MT)$ программы любой машины Тьюринга MT перерабатывает в однобуквенное слово 0, если машина MT несамоприменима, или в однобуквенное слово 1, если машина MT самоприменима. То есть $\Theta[D(MT)] = 0$ для несамоприменимых MT и $\Theta[D(MT)] = 1$ для самоприменимых MT.

Если машина Тьюринга Θ существует, то можно немного изменить ее программу так, чтобы вместо остановки с оставлением на ленте однобуквенного слова 1 она зацикливалась бы. Значит, если машина Тьюринга Θ существует, то существует и машина Тьюринга Ω, которая применима к записи программы произвольной машины Тьюринга MT тогда и только тогда, когда машина MT несамоприменима (в этом случае $\Omega[D(MT)] = 0$). Если же машине Ω подается на вход запись программы самоприменимой машины MT, то машина Ω зацикливается.

Самоприменима ли машина Ω? Если да, то, как всякая самоприменимая машина, она не должна зациклиться, получив на вход запись собственной программы. Но, в силу своего определения, она должна зациклиться, получив на вход запись собственной программы, ибо она должна всегда зацикливаться, получая на вход записи программ любых самоприменимых машин. Это противоречие показывает, что машина Ω не может быть самоприменимой. Значит, она несамоприменима. Если это так, то, как всякая несамоприменимая машина, она должна зациклиться, получив на вход запись собственной программы. Но, в силу своего определения, она не должна зациклиться и даже должна выдать на выходе однобуквенное слово 0, ибо так она должна всегда делать, получая на вход записи программ несамоприменимых машин. Полученное теперь противоречие показыва-

ет, что машина Ω не может быть несамоприменимой. Получен парадокс: машина Ω не может быть ни самоприменимой, ни несамоприменимой!

Единственный выход из сложившейся парадоксальной ситуации состоит в том, чтобы признать: машины Ω не существует. Но она должна была бы существовать, если бы существовала машина Θ, модификацией которой получена машина Ω. Значит, машины Θ не существует. А именно это мы и хотели доказать.

В согласии с тезисом Тьюринга, теперь мы можем заявить, что и вообще не существует никакого алгоритма, распознающего самоприменимость машин Тьюринга. Иначе говоря, *проблема самоприменимости* оказывается *алгоритмически неразрешимой* (невзирая на то, что вообще она разрешима).

Алгоритмическая неразрешимость проблемы применимости. Теперь может быть установлена алгоритмическая неразрешимость *проблемы распознавания применимости для машин Тьюринга*, которая состоит в следующем: пусть заданы функциональная схема (программа) какой-нибудь машины Тьюринга и подаваемое на вход машины слово. Требуется узнать, применима ли машина к данному слову.

Ясно, что если бы существовал алгоритм для решения этой проблемы, то с его помощью можно было бы узнать, применима ли машина к слову, кодирующему ее собственную программу, то есть самоприменима ли она. Но нам уже известно, что проблема самоприменимости алгоритмически неразрешима. Значит, и проблему распознавания применимости для машин Тьюринга следует тоже признать алгоритмически неразрешимой.

Значение теории машин Тьюринга для искусственного интеллекта. Самим Тьюрингом теория машин Тьюринга была предложена в первую очередь как инструмент исследования поставленной Гильбертом проблемы разрешимости. Гильберт полагал, что для всякой разрешимой задачи обязательно существует и алгоритм, решающий эту задачу. В контексте искусственного интеллекта как науки о создании мыслящих машин справедливость гипотезы Гильберта означала бы, что одной только алгоритмической модели представления знаний полностью достаточно для достижения целей искусственного интеллекта.

Гипотеза Гильберта, однако, несправедлива. Алгоритмически неразрешимые задачи существуют. И особенно неприятно то, что даже задачи, *вообще разрешимые* (то есть доступные для решения мыслящим субъектом), могут быть *неразрешимы алгоритмически* (то есть в рамках одной только алгоритмической модели представления знаний никогда не будут решены мыслящими по этой модели машинами). Алгоритмически мыслящий искусственный интеллект никогда не достигнет некоторых из тех вершин, которых способен достичь интеллект естественный. А это значит, что нельзя сводить искусственный интеллект к одной только алгоритмической модели представления знаний. Существование других моделей (логической, продукционной, фреймовой, синаптической) есть не творческий произвол исследователей в области искусственного интеллекта, а результат осознания не-

обходимости, продиктованной теми следствиями, которые вытекают из несправедливости гипотезы Гильберта.

С другой стороны, широкие классы алгоритмически разрешимых задач все-таки существуют, и задачи этих классов могут представлять значительный теоретический и практический интерес. Язык машин Тьюринга является очень простым по используемым им «правилам игры» и весьма мощным по своим выразительным возможностям средством точного описания алгоритмов (то есть их формализации). Ясно, что создание мыслящих машин (пусть и с ограниченными, но широкими и значимыми универсумами мысли, характеризующимися алгоритмической разрешимостью возникающих в них задач) без формализации алгоритмов является невозможным.