Лекция 16

Машины Тьюринга. Тезис Чёрча – Тьюринга

Примеры невычислимых функций и неразрешимых множеств, которые мы строили выше, определяются через универсальные вычислимые функции. Эти примеры лежат в основе всех наших знаний о неразрешимости. Однако для доказательства неразрешимости конкретных множеств нужно больше — требуется построить явную конструкцию какой-нибудь универсальной вычислимой функции. А для этого нужно дать явное определение алгоритма. При этом нужно иметь в виду, что чем проще определение, тем легче его использовать в доказательствах.

16.1 Машины Тьюринга

Мы рассмотрим классическую модель вычислений, на которой будут основано точное определение вычислимых функций, — *машины Тьюринга* (МТ).

МТ состоит из

- бесконечной в две стороны ленты, в ячейках которой могут быть записаны символы *алфавита* A (некоторого конечного множества);
- *головки*, которая может двигаться вдоль ленты, обозревая в каждый данный момент времени одну из ячеек;
- *оперативной памяти*, которая имеет конечный размер (другими словами, состояние оперативной памяти — это элемент некоторого конечного множества, которое называется *множеством состояний МТ* Q);
- *таблицы переходов* (или программы), которая задаёт функцию

$$\delta\colon A \times Q \to A \times Q \times \{-1, 0, +1\}.$$

Поскольку таблица переходов — это функция на конечном множестве, её возможно задать таблицей. Каждая строка таблицы — это пять значений

$$a,\ q,\ a',\ q', d, \quad \text{(другие способы записи: } \delta(a,q) = (a', q', d) \text{ или } \delta\colon (a,q) \mapsto (a', q', d)).$$

которые описывают следующий порядок действий МТ: если головка МТ находится над ячейкой, содержащей символ a, а состояние МТ равно q, то на очередном такте работы МТ записывает в текущую ячейку символ a', изменяет состояние на q' и сдвигает головку на d ячеек (отрицательное значение отвечает сдвигу влево, положительное — сдвигу вправо). Пример такта работы МТ изображен на рис. 16.1.

Работа МТ состоит из последовательного выполнения тактов в соответствии с таблицей переходов. Может так случиться, что для текущей пары значений (a, q) функция переходов не определена. В этом случае работа машины заканчивается (машина останавливается). Обычно среди состояний МТ выделяют множество Q_f *финальных состояний* — таких состояний q_f, что таблица переходов не определена для всех пар (a, q_f). Попав в финальное состояние, машина обязательно остановится, отсюда и название.

Лента МТ бесконечна и это не соответствует нашей интуиции об алгоритмах: алгоритм на каждом шаге работы оперирует лишь данными конечного размера. Чтобы учесть это обстоятельство, мы предполагаем, что в алфавите машины есть специальный символ Λ (пробел или *пустой символ*) и все ячейки ленты за исключением конечного числа содержат пустые символы. Это свойство ленты сохраняется при работе МТ, поскольку за такт работы меняется содержимое не более одной ячейки ленты.

Состояние такой машины уже описывается конечными данными. Вместо картинок как на рис. 16.1 мы будем использовать *конфигурации*. Конфигурация — это слово в алфавите $A \cup Q$, в котором первый и последний символы непустые, и ровно один символ принадлежит множеству состояний. Договоримся считать, что символ состояния записывается слева от символа в той ячейке, над которой находится головка МТ. На ленте слева и справа от символов конфигурации стоят только пустые символы. На рис. 16.2 проиллюстрировано соответствие между лентой с головкой и конфигурациями.

Мы также предполагаем, что работа МТ начинается из конфигурации вида $q_0 u$, где q_0 — особое *начальное* состояние машины, а слово u — входные данные (или *вход*) машины. Конфигурации МТ преобразуются такт за тактом, порождая после-

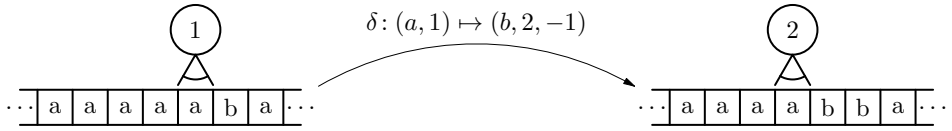

Рис. 16.1: Такт работы машины Тьюринга

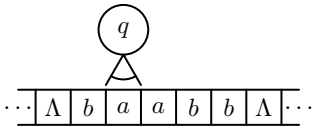

Рис. 16.2: Конфигурация МТ: слово *bqaabb*

довательность конфигураций

$$c_0 = q_0u,\ c_1,\ c_2, \ldots, c_t,\ \ldots$$

Эта последовательность бесконечна, если машина не останавливается, и конечна в противном случае. *Результатом* работы является та часть финальной конфигурации, которая расположена между символом состояния и ближайшим к нему пустым символом справа (см. рис. 16.3).

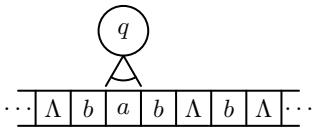

Рис. 16.3: Результат МТ: слово *ab*

Определение 16.1. МТ M *вычисляет* функцию $f\colon B^* \to B^*$ (где B — подмножество алфавита машины, не содержащее пустого символа), если для каждого w из области определения функции f результат работы M равен $f(w)$, а для каждого w не из области определения f машина M не останавливается на входе w.

Функция f называется вычислимой машинами Тьюринга, если есть такая МТ, которая вычисляет f.

Замечание 16.1. В литературе встречаются различные определения машин Тьюринга и функций, вычислимых машинами Тьюринга. Эти различия несущественны, хотя от деталей определений зависят рассуждения о МТ. Всюду далее мы предполагаем данные выше определения.

Пример 16.1. Рассмотрим машину M_b с алфавитом A, множеством состояний $\{0, 1\}$, где 0 — начальное состояние, и таблицей переходов

$$\delta\colon (a, 0) \mapsto \begin{cases} (a, 0, +1), & \text{если } a \neq b, \\ (b, 1, 0), & \text{если } a = b. \end{cases}$$

Это описание машины, которое задаёт её однозначно. И ничего больше — в описании не содержится никаких утверждений о такой машине

Мы утверждаем, что такая машина переводит головку на ближайший справа символ b и после этого останавливается в состоянии 1 (и не останавливается, если справа нет ни одного символа b). Справедливость такого утверждения нужно доказывать.

Заметим, что если головка находится над символом b, то машина остановится после первого такта: это сразу следует из описания таблицы переходов.

Если же машина находится над каким-то другим символом, то из таблицы переходов видно, что головка сдвигается вправо без изменения состояния и символа в ячейке. Формальное завершение этого рассуждения требует применения индукции (по количеству символов между начальным положением головки и ближайшим справа символом b).

Машину из примера легко модифицировать, чтобы перемещать головку влево до первого символа b. Дальше в описании более сложных машин мы будем ссылаться на машину M_b и её левый аналог как на инструкцию «сдвинуться вправо до символа b» («сдвинуться влево ...»).

Пример 16.2. Построим машину, которая вычисляет функцию $f\colon w \mapsto wa$ к слову в алфавите $\{a\}$.

Зададим машину таблицей переходов (0 — начальное состояние):

$$\delta\colon\ \mapsto \begin{cases} (a,0) \mapsto (a,0,+1) \\ (\Lambda,0) \mapsto (a,1,-1) \\ (a,1) \mapsto (a,1,-1) \\ (\Lambda,1) \mapsto (\Lambda,2,+1). \end{cases}$$

Докажем корректность (т.е. что эта машина и впрямь вычисляет заданную функцию). Аналогично предыдущему примеру доказывается, что конфигурацию $0a^n$ машина переводит в конфигурацию a^n0 (исполняется только первая строчка таблицы переходов). На следующем такте работы исполняется вторая строчка и конфигурация становится $a^{n-1}1a^2$, если $n > 0$, и $1\Lambda a$, если $n = 0$. В первом случае на последующих шагах работы исполняется третья строчка до тех пор, пока машина не увидит символ Λ, т.е. получится конфигурация $1\Lambda a^{n+1}$ (это утверждение легко доказывается по индукции). Теперь исполняется четвёртая строчка и в конфигурации $2a^{n+1}$ машина заканчивает работу, так как из описания машины следует, что 2 — финальное состояние.

Результат работы такой машины по определению равен $a^{n+1} = a^n a$.

16.2 Тезис Чёрча – Тьюринга

Тезис Чёрча – Тьюринга: *всякая вычислимая функция вычислима машиной Тьюринга.*

Другими словами, машина Тьюринга является формальным определением понятия алгоритма.

Тезис Чёрча – Тьюринга не является математическим утверждением. Это математическое определение. Его можно также воспринимать как закон природы: утверждение об окружающем нас мире, основанное на опыте.

Обсудим неформальные обоснования для этого тезиса. Во-первых, почему работа МТ реализуется алгоритмом? Это почти очевидно: каждый такт работы МТ состоит в выполнении очень простого набора действий: найти в таблице переходов строчку, которая начинается с текущего символа и текущего состояния; если таковой не нашлось, закончить работы и выделить из конфигурации результат; в противном случае заменить текущий символ на тот, который указан в найденной строчке, изменить состояние на указанное в найденной строчке, переместить головку влево, оставить на месте или переместить вправо в зависимости от указанной в найденной строчке команды движения. Фактически мы описали простые и однозначно понимаемые инструкции действий, что и есть алгоритм в неформальном понимании слова.

В другую строну тезис куда менее очевиден. Мы знаем алгоритмы, которые работают со сложными структурами данных и действия которых организованы куда более сложным образом, чем описанная выше циклическая последовательность локальных замен в слове. Почему ни один такой алгоритм не даёт больше возможностей, чем машина Тьюринга?

Объяснение, восходящее к Тьюрингу, тут такое. Любой алгоритм потенциально может быть исполнен человеком. Конечно, время такого исполнения может оказаться намного больше времени человеческой жизни, но нас интересует лишь потенциальная возможность.

А исполнение алгоритма человеком в предельно упрощенном виде можно представить так. У человека есть карандаш, ластик, неограниченный запас листов бумаги и книжечка с инструкциями (собственно алгоритм). Бумага лежит в двух стопках, человек может выполнять действия лишь на одной стороне листа. Это аналог ленты машины. Действия определяются номером страницы в книге инструкций (состояние МТ) и содержимым верхнего листа бумаги (символ алфавита в ячейке, на которую смотрит головка). Чтобы действие было однозначно понимаемым, разных содержимых листа бумаги должно быть конечное число. Вот таким образом и приходим к формальному определению.

Если оставаться на чисто математических позициях, то нужно рассуждать так: у нас появилось определение алгоритма, так что теперь нужно его использовать для доказательства утверждений. В частности, все утверждения, которые мы раньше доказывали в рамках «наивной» теории алгоритмов, нужно теперь доказать аккуратно на основе определения алгоритма как машины Тьюринга.

16.3 Использование машин Тьюринга в доказательствах

Насколько подробно нужно описывать машину Тьюринга в рассуждении? Это зависит от наших целей. Если кому-то захочется программировать на машинах Тьюринга, то придётся описывать машину в соответствии с данными выше определениями (указать алфавит, множество состояний, таблицу переходов и т.д.). Однако само по

себе такое описание не является доказательством какого-либо утверждения.

А если нас интересует лишь существование машины, которая выполняет то или иное преобразование, то достаточно описать машину с той степенью подробности, которая нужна в доказательстве корректности её работы.

Для облегчения доказательств корректности МТ есть несколько приёмов.

Блочная структура. Как обычно в программировании, при описании МТ удобно разбивать всю программу (в данном случае таблицу переходов) на блоки (подпрограммы, процедуры и т.п.)

Простейший вариант разбиения на блоки — это последовательное соединение машин. Опишем последовательное соединение двух машин, случай нескольких машин аналогичен.

Пусть есть МТ M_1 и M_2. Их последовательное соединение — это МТ, которая получается после следующих преобразований таблиц переходов машин M_1 и M_2:

– переименовываем состояния машин так, чтобы они не пересекались (легко видеть, что от переименования состояний результат работы машины не меняется);

– для каждой пары (a, q), на которой не определена таблица переходов машины M_1, добавляем в таблицу переходов машины–соединения строчку

$$\delta(a, q) = (a, q_0^{(2)}, 0),$$

здесь $q_0^{(2)}$ — начальное состояние машины M_2.

Таким образом, если машина M_1 останавливается, то обязательно начинает работу машина M_2. (Контрольный вопрос, почему необходимо переименование состояний?)

Метки на ленте. Алфавит МТ конечен, но если нас интересует лишь существование машины, он может быть сколь угодно велик. Этим обстоятельством удобно пользоваться при описании машин, говоря о «метках» на ячейках ленты. Формально использование меток состоит в том, что мы заменяем алфавит A на расширенный алфавит $A \times L$, где L — то вспомогательное множество меток. Договоримся, что среди меток всегда есть пустой символ Λ, который обозначает отсутствие метки. Символы исходного алфавита отождествим с парами (a, Λ), это позволяет говорить о вычислении функций в алфавите, который содержится в исходном алфавите A. В таблице переходов МТ метки позволяют определять команды вида «если данная ячейка помечена, сделай то-то, а если нет, то сделай иное».

Структурирование оперативной памяти. Аналогично алфавиту удобно расширять и множество состояний, разделив его на две части: «управляющие состояния» и «оперативная память». Управляющие состояния используются для организации условных переходов и соединения машин (как это было описано выше). Оперативная память используется для хранения информации. Скажем, машине потребовалось «запомнить» символ текущей ячейки. Для этого она меняет состояние на пару («управляющее состояние», «значение символа»).

Важно помнить, что множество состояний МТ конечно. Поэтому и «оперативная память» конечна — машина в состоянии запомнить лишь элемент какого-то конечного множества.

16.4 Композиция функций, вычислимых МТ, и уборка мусора

После этих предварительных соглашений можно перейти к доказательствам утверждений о МТ.

Как мы помним, одно из важных свойств вычислимых функций — замкнутость относительно композиции. Докажем это свойство для функций, вычислимых машинами Тьюринга.

Теорема 16.1. *Пусть $f\colon B^* \to B^*$, $g\colon B^* \to B^*$ вычислимы МТ. Тогда $f \circ g$ также вычислима МТ.*

Как мы и говорили выше, результат работы одного алгоритма можно подать на вход другого алгоритма.

Поэтому предложим такое рассуждение. Пусть M_1 вычисляет g, а M_2 вычисляет f. Тогда МТ, которая состоит из последовательного соединения блоков M_1 и M_2 вычисляет $f \circ g$.

К сожалению, это рассуждение неверно. Дело в том, как определен результат вычисления МТ. Помимо собственно результата на ленте могут оказаться посторонние символы — «мусор» — и они изменят работу машины M_2. Поэтому такое соединение машин, вообще говоря, вычисляет какую-то другую функцию, а не композицию функций f и g.

Тем не менее, если первая машина M_1 заканчивает работу, оставляя на ленте только полезный результат и ничего больше, рассуждение становится корректным. Поэтому для завершения доказательства теоремы 16.1 нам нужно решить проблему «уборки мусора».

Заметим, что убирать мусор в конце работы поздно: непустые символы могут быть отделены от текущего положения головки сколь угодно длинными последовательностями из пустых символов.

Задача 16.3. Докажите, что не существует МТ с таким «волшебным состоянием» q_m, что любую конфигурацию, содержащую q_m, машина переводит в пустую конфигурацию q_f, где q_f — финальное состояние.

Решить проблему уборки мусора можно, предусмотрев заранее некоторые санитарные процедуры. А именно, добавим к алфавиту еще два символа $\triangleleft$, $\triangleright$, которые будут ограничивать рабочую зону на ленте (в процессе работы машины слева от $\triangleleft$ и справа от $\triangleright$ всегда будут только пустые символы). При разрастании рабочей зоны эти символы нужно сдвигать. В конце работы нужно будет выделить результат и стереть (заменить на пустые) все символы внутри рабочей зоны, которая ограничена символами $\triangleleft$, $\triangleright$.

Лемма 16.2 (об уборке мусора)**.** *Пусть машина M вычисляет функцию f. Тогда существует такая машина M', которая вычисляет ту же функцию, но финальная конфигурация которой на любом входе w из области определения f имеет вид $q_f f(w)$.*

Доказательство. Машина M' будет последовательным соединением четырёх машин.

Первая машина M_1 преобразует начальную конфигурацию $q_0' w$ в конфигурацию $\triangleleft q_0 w \triangleright$, где q_0 — начальное состояние машины M.

Вторая машина M_2 работает так же, как исходная машина M, но сохраняет окаймление конфигурации символами $\triangleleft$, $\triangleright$.

Третья машина M_3 стирает символы слева от положения головки в финальной конфигурации машины M_2.

Четвёртая машина M_4 стирает символы справа от последнего символа результата работы M_2 и возвращает головку в ту ячейку, в которой она была при остановке машины M_2.

Как ясно из этого описания, при корректной реализации каждой из этих четырёх машин их соединение M' удовлетворяет искомому свойству.

Опишем реализации этих четырёх машин.

Машина M_1 последовательно выполняет следующие шаги:

1. сдвинуться на одну ячейку влево, записать в неё символ $\triangleleft$;
2. сдвинуться до первого пустого символа справа;
3. записать символ $\triangleright$;
4. сдвинуться до символа $\triangleleft$ слева;

Шаги 1, 3, 4 требуют выполнения конечного числа действий и потому реализуемы подходящей МТ. Шаг 2 исполняется МТ из примера 16.1. Корректность такой машины очевидна.

Таблица переходов машины M_2 совпадает с таблицей переходов исходной машины M за исключением работы на добавленных символах $\triangleleft$, $\triangleright$. На символе $\triangleleft$ машина выполняет следующие такты работы:

1. записать пустой символ, сдвинуться влево и перейти в состояние $\tilde{q}$;
2. записать символ $\triangleleft$, сдвинуться вправо перейти в состояние q.

На рис. 16.4 показано выполнение этих шагов. Работа M_2 на символе $\triangleright$ устроена аналогично (разумеется, символ переносится вправо).

Как видно из описания, количество состояний у машины M_2 в два раза больше, чем у исходной машины M, а работа при чтении добавленных символов окаймления устроена так, что машина переносит символ окаймления на одну ячейку вне рабочей зоны, возвращается назад и продолжает работу как исходная машина M. Поэтому машина M_2 закончит работу, имея слева и справа от рабочей зоны символы $\triangleleft$, $\triangleright$.

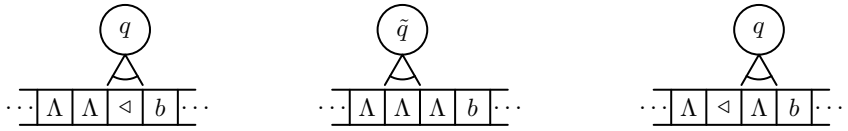

Рис. 16.4: Сдвиг левого ограничителя рабочей зоны

Опишем теперь устройство машины M_3. Она начинает работу в одном из финальных состояний q_f исходной машины M. Работа машины M_3 разбивается на следующие шаги:

1. пометить текущую ячейку, сдвинуться влево и перейти в состояние q_ℓ;
2. идти влево до символа $\triangleleft$, заменяя символ в каждой ячейке на пустой символ;
3. на символе $\triangleleft$ записать пустой символ и перейти в финальное состояние q_r.

Второй шаг реализуется такой таблицей переходов:

$$\delta(a, q_\ell) = (\Lambda, q_\ell, -1), \quad a \neq \triangleleft.$$

Ясно, что в результате работы машины M_3 все символы слева от результата работы исходной машины M будут заменены на пустые.

Последняя машина M_4 должна убирать мусор справа от результата работы исходной машины. Она начинает работу в состоянии q_r (финальном состоянии предыдущей машины) и исполняет следующие шаги:

1. сдвинуться вправо до помеченной ячейки;
2. сдвинуться вправо до пустого символа (быть может, помеченного);
3. идти вправо до символа $\triangleright$, заменяя символ в каждой ячейке на пустой символ;
4. на символе $\triangleright$ записать пустой символ;
5. идти влево до помеченной ячейки;
6. убрать пометку и остановиться.

Шаги 1, 2, 5 исполняются аналогично примеру 16.1. (Контрольный вопрос, почему необходим первый шаг?) Шаг 3 исполняется аналогично шагу 2 машины M_3. Реализация остальных шагов очевидна. Обратите также внимание на уточнение в шаге 2. Оно необходимо для корректности работы машины в случае пустого результата работы (машина M_3 в этом случае ставит пометку на пустой символ). □

Доказательство теоремы 16.1. По лемме об уборке мусора каждая вычислимая на МТ функция вычислима машиной, которая в конце работы оставляет на ленте только результат работы.

Последовательное соединение таких машин вычисляет композицию функций, вычислимых этими машинами. □

16.5 Многоленточные машины Тьюринга

Как уже ясно из названия, у многоленточных машин не одна лента, а несколько (фиксированное число для конкретной машины). На каждой ленте есть своя головка. За такт работы головки могут перемещаться по всем лентам. Действие на такте работы зависит как от состояния машины, так и от всего набора символов, которые видят головки машины на всех лентах.

Как эти неформальные изменения отражаются в формальном определении многоленточной машины и функций, вычисляемых такими машинами? Чтобы задать машину с h лентами, нужно указать:

– алфавит A, в котором выделен пустой символ Λ;

– множество состояний Q, в котором выделено начальное состояние q_0;

– таблицу переходов, которая теперь является функцией вида

$$\delta\colon A^h \times Q \to A^h \times Q \times \{-1, 0, +1\}^h$$

(первый аргумент — символы, которые машина видит на ленте; последний — команды движения для головок на каждой ленте);

– выделить среди лент *ленту входа* и *ленту результата* (возможно, что это одна и та же лента).

Таблица переходов по прежнему является функцией на конечном множестве, поэтому её возможно задать таблицей.

Работа МТ состоит из последовательного выполнения тактов в соответствии с таблицей переходов. Может так случиться, что для текущего набора значений $(a_1, a_2, \ldots, a_h, q)$ функция переходов не определена. В этом случае работа машины заканчивается (машина останавливается). Как и раньше, можно ввести множество финальных состояний Q_f, т.е. тех состояний q_f, для которых таблица переходов не определена для всех значений $(a_1, a_2, \ldots, a_h, q_f)$. В финальном состоянии машина обязательно останавливается.

Мы предполагаем, что h-МТ начинает работу в состоянии q_0, а все ленты кроме ленты входа содержат только пустые символы. На ленте входа записано входное слово, и головка находится над первой слева ячейкой, содержащей символы этого слова.

Поскольку за такт работы меняется содержимое не более одной ячейки ленты, в процессе работы машины на каждой ленте будет записано лишь конечное количество непустых символов.

Конфигурация многоленточной машины может быть задана набором конфигураций на каждой ленте

$$(u_1 q v_1,\ u_2 q v_2,\ \ldots,\ u_h q v_h).$$

Символ состояния один и тот же, так как по нашим определениям состояние есть у машины, а не у головки.[1]

Далее нам будет удобен другой способ представления конфигурации машины. Выровняем ленты и будем рассматривать «окно», в которое заведомо помещаются все непустые символы на каждой ленте. В таком случае конфигурация однозначно определяется матрицей размера $h \times N$, в которой записаны символы на лентах. Нужно ещё указать положения головок на лентах (они-то не обязательно выровнены — машина способна перемещать головки независимо). По этой причине будем помещать в матрицу не символы алфавита A, а пары $(a, \hat{q})$, где $\hat{q}$ указывает, расположена ли на данной ленте головка над данной ячейкой. Если да, то $\hat{q} \in Q$ — текущее состояние машины. Если нет, то $\hat{q}$ — какой-то символ не из Q, который указывает, что над над данной ячейкой на данной ленте нет головки. Будем для единообразия использовать в качестве такого символа Λ. Такую матрицу в дальнейшем называем *матрицей конфигурации*.

Вот пример матрицы начальной конфигурации для двухленточной машины:

(Λ, q_0)	(Λ, Λ)	(Λ, Λ)	(Λ, Λ)	(Λ, Λ)
(a, q_0)	(b, Λ)	(a, Λ)	(a, Λ)	(b, Λ)

Как и для одноленточной машины, работа h-МТ порождает последовательность конфигураций

$$c_0 = q_0 u,\ c_1,\ c_2, \ldots, c_t,\ \ldots$$

Эта последовательность бесконечна, если машина не останавливается, и конечна в противном случае. *Результатом* работы является та часть финальной конфигурации на ленте результата, которая расположена между положением головки и ближайшим к нему пустым символом справа. Например, если у двухленточной МТ лента результата — нижняя, то результатом работы МТ, остановившейся в конфигурации, заданной окном

(Λ, Λ)	(Λ, Λ)	(a, q_f)	(a, Λ)	(Λ, Λ)
(a, Λ)	(b, q_f)	(a, Λ)	(Λ, Λ)	(b, Λ)

,

будет ba.

Определение функции, вычислимой h-МТ, сохраняется дословно.

Определение 16.2. h-МТ M *вычисляет* функцию $f\colon B^* \to B^*$ (где B — подмножество алфавита машины, не содержащее пустого символа), если для каждого w из области определения функции f результат работы M равен $f(w)$, а для каждого w не из области определения f машина M не останавливается на входе w.

Многие алгоритмические действия выполняются на многоленточных машинах проще, чем на одноленточных. Предлагаем читателю в этом убедиться, решив следующие задачи.

[1] Разумеется, возможно и такое определение, в котором состояния головок различаются. В этом случае определение таблицы переходов нужно изменить (подумайте, каким именно образом); но полученная модель вычислений будет эквивалентна нашей.

Задача 16.4. Постройте двухленточную машину, которая (а) копирует с одной ленты на другую символы от текущего положения головки до ближайшего справа символа-разделителя #, скопированное слово на второй ленте начинается с первоначального положения головки на ней; (б) переносит указанные символы (аналогично предыдущему, но на первой ленте символы заменяется на пустые).

Задача 16.5. Постройте двухленточную машину, которая сравнивает слова на двух лентах от текущего положения головок до символа-разделителя. В случае равенства слов машина заканчивает работу в состоянии q_y, в случае неравенства — в состоянии q_n.

Описанные в этих задачах действия легко модифицировать, если концом зоны, которую нужно скопировать (перенести или сравнить) является не один символ-разделитель, а какое-нибудь фиксированное слово или даже фиксированный набор слов.

16.6 Моделирование многоленточной МТ на одноленточной

Теорема 16.3. *Любая функция, вычислимая на многоленточной МТ, вычислима и на одноленточной машине.*

Доказательство теоремы состоит в том, что по многоленточной машине строится одноленточная машина, которая моделирует работу многоленточной.

Чтобы понять идею такого моделирования, рассмотрим описание конфигурации многоленточной машины M_h в виде матрицы конфигурации размера $h \times N$. Каждый столбец такой матрицы может находиться в конечном числе состояний.

Задача 16.6. Проверьте, что различных состояний столбца матрицы конфигурации h-МТ не более $(A \cdot (Q+1))^h$, где A — размер алфавита, а Q — количество состояний.

Моделирующая машина M_1 использует расширенный алфавит из $A+(A\cdot(Q+1))^h$ (пустой символ и символы, отвечающие различным столбцам матрицы конфигурации). Она поддерживает описание матрицы конфигурации машины M_h в этом алфавите и изменяет его, моделируя работу M_h по тактам.

Поскольку действия M_h на каждом такте работы зависят от её состояния и символов под головками на каждой ленте, машина M_1 поддерживает также и эту информацию, записывая её в «оперативную память». Т.е. состояния M_1 представляются парами («управляющее состояние», «оперативная память»).

Такт работы машины M_h моделируется машиной M_1 в два этапа.

На первом этапе машина M_1 просматривает все непустые ячейки на своей ленте слева направо и определяет, какие символы расположены под текущими положениями головок машины M_h.

На втором этапе M_1 изменяет содержимое своей ленты в соответствии с таблицей переходов машины M_h.

Опишем более детально устройство машины M_1. Алфавит машины M_1 — это множество

$$A' = A \cup (A \times (Q \cup \Lambda))^h,$$

пустой символ тот же, что и у моделируемой машины M_h, — Λ.

Машина M_1 является последовательным соединением трёх машин.

Первая машина M_s подготавливает содержимое ленты к двухэтапному моделированию тактов работы машины M_h. Машина M_s просматривает ячейки входного слова. Первый символ a_1 она заменяет на $((a_1, q_0), (\Lambda, q_0), \ldots, (\Lambda, q_0))$ (это перввй столбец матрицы начальной конфигурации M_h), а каждый последующий символ входа a на $((a, \Lambda), (\Lambda, \Lambda), \ldots, (\Lambda, \Lambda))$ (это остальные столбцы матрицы начальной конфигурации — напомним, что в начальной конфигурации все ленты кроме входной пусты). Обнаружив пустой символ Λ, машина M_s возвращается в крайнее левое положение и останавливается.

Вторая машина M_w моделирует такты работы M_h описанным выше способом. Она проходит непустые ячейки ленты два раза. При движении слева направо машина M_w «запоминает» символы под головками машины M_h по следующему правилу: если в очередном столбце матрицы конфигурации машины M_h на i-й позиции находится пара (a, q), $q \in Q$, то i-я головка расположена над символом a.

К концу первого прохода в оперативной памяти M_w содержится полная информация о символах под головками и состоянии M_h, что однозначно определяет строчку таблицы переходов M_h, которую нужно применить на данном такте. Если такой строчки нет, то M_w заканчивает работу.

На втором проходе найденная строчка таблицы переходов M_h используется для обновления матрицы конфигурации. Информация о символах на лентах M_h обновляется по следующему правилу: если в очередном столбце матрицы конфигурации машины M_h на i-й позиции находится пара (a, q), $q \in Q$, то столбец меняется так, чтобы в этой позиции было написана пара (a', q), где a' — символ, который M_h записывает на i-ую ленту. Те пары в столбце, которые соответствуют ячейкам, над которыми нет головки, не изменяются.

Кроме того нужно обновить информацию о положениях головок соответственно текущей команде движения. Для этого машина M_w переписывает вторые компоненты пар, из которых состоит столбец матрицы конфигурации M_h (т.е. текущий столбец матрицы). Движение по каждой ленте может быть как влево, так и вправо. Поэтому для выполнения этого действия M_w перемещается из текущего положения на шаг вправо, записывая в этот столбец новые положения головок, и на шаг влево, выполняя аналогичное действие. При выполнении этих действий машина M_w может выйти за пределы рабочей зоны (матрицы конфигурации). Тогда она оказывается над пустым символом, который заменяется на подходящий столбец, описывающий пустые символы и положения головок машины M_h.

По завершении работы M_w начинает работу третья машина M_f, которая восстанавливает на ленте состояние ленты результата машины M_h. Она проходит по всем ячейкам рабочей зоны и заменяет столбец матрицы конфигурации на символ из алфавита A машины M_h, если головка M_h на ленте результата не находится в

этом столбце. Затем она возвращается в ту ячейку, которая соответствует положению головки на ленте результата машины M_h, и производит ту же замену. После этого M_f останавливается с чувством выполненного долга.

16.7 Универсальная машина Тьюринга

Важнейшим свойством вычислимых функций является существование универсальной вычислимой функции. Выбрав в качестве формального определения алгоритма машины Тьюринга, мы получаем класс функций, вычислимых машинами Тьюринга. Нужно доказать, что этот класс функций обладает желаемыми свойствами. В частности, нужно доказательство существования универсальной машины.

Используя неформальное обоснование тезиса Чёрча – Тьюринга, легко убедить себя в том, что универсальная машина существует. Однако более детальное доказательство этого факта само по себе даёт дополнительную уверенность в тезисе Чёрча – Тьюринга и позволяет пересказывать результаты о неразрешимости, аналогичные тем, что были получены в абстрактной постановке.

Самый короткий путь к построению универсальной машины лежит через построение многоленточной универсальной машины. Как мы уже знаем, многоленточные машины моделируются одноленточными, так что этого достаточно.

По аналогии с абстрактной теорией хотелось бы сказать, что универсальная машина Тьюринга U моделирует работу любой другой машины в том смысле, что получив на вход описание МТ M и описание её входа x, она выдаёт тот же результат $M(x)$.

Однако такая машина невозможна: ведь её алфавит конечен, а существуют МТ со сколь большим алфавитом. Поэтому нужно научиться описывать и МТ, и их входные слова в каком-то одном алфавите. После этого уже можно сформулировать требования к универсальной машине.

Описание машин и их входов в одном алфавите — несложная задача, хотя и требующая утомительных подробностей.

Зафиксируем формат описания МТ и входных слов. Будем считать, что и алфавит, и множество состояний — это отрезки натурального ряда:

$$A = \{0, 1, \ldots, n\}, \quad Q = \{0, 1, \ldots, k\},$$

причём для единообразия полагаем начальное состояние $q_0 = 0$ и пустой символ $\Lambda = 0$. Таблица переходов — это множество пятёрок вида

(⟨старый символ⟩, ⟨старое состояние⟩, ⟨новый символ⟩, ⟨новое состояние⟩, ⟨команда движения⟩).

В команде движения будем считать, что 0 отвечает сдвигу влево, 2 — сдвигу вправо, а 1 означает, что головка должна остаться в прежнем положении.

Для простоты числа мы будем кодировать равномерно: записи всех символов имеют одинаковую длину:

$$\langle i \rangle = \underbrace{00\ldots0}_{i \text{ символов}}\,1\,\underbrace{00\ldots00000}_{n-i \text{ символов}}.$$

Строка таблицы переходов будет задаваться пятью кодами такого вида, разделёнными символом #.

Итак, описание $\langle M\rangle$ МТ M в таком формате выглядит как набор записей вида

$$\#\langle a\rangle\#\langle q\rangle\#\langle a'\rangle\#\langle q'\rangle\#\langle d\rangle\# \ .$$

Входное слово кодируется аналогично как последовательность кодов входящих в него символов:

$$\#\langle a_1\rangle\#\langle a_2\rangle\#\langle a_3\rangle\#\ldots\#\langle a_n\rangle\# \ .$$

Пустое слово будем кодировать как $\#10^n\#$ (код пустого символа).

Таким образом, для описания МТ и их входов мы используем алфавит $\{0,1,\#\}$. В абстрактной теории алгоритмов мы рассматривали универсальную функцию $U(p,x)$ от пар чисел (программа вычисления функции одного аргумента, аргумент). Теперь нам удобнее рассматривать функции от пар слов в алфавите $\{0,1,\#\}$, что не изменяет абстрактной теории благодаря существованию вычислимой биекции между словами и натуральными числами, как это уже объяснялось раньше.

Универсальной будем называть функцию $U\colon \{0,1,\#\}\times\{0,1,\#\}\to\{0,1,\#\}$, значение которой на входах $(\langle M\rangle,\langle x\rangle)$, где $\langle M\rangle$ — описание машины M, а $\langle x\rangle$ — описание входного слова x, равно $\langle M(x)\rangle$.

Универсальная машина (УМТ) должна вычислять какую-нибудь универсальную функцию. Мы ещё не определили, как МТ вычисляет функции от нескольких аргументов. Наиболее естественный способ — разделять аргументы каким-нибудь специальным символом $\odot$.

Определение 16.3. МТ M *вычисляет* функцию $f\colon B^*\times B^*\to B^*$, если для каждой пары (u,v) из области определения функции f результат работы M на входе $u\odot v$ равен $f(u,v)$, а для каждой пары (u,v) не из области определения f машина M не останавливается на входе $u\odot v$.

Аналогично определяется вычисление функции от нескольких аргументов.

Замечание 16.2. Обратите внимание, что поведение универсальной функции определено только на входах, которые являются кодами машины и входного слова.

Если есть УМТ, которая вычисляет какую-нибудь универсальную функцию на таких входах, она также как-то работает и на остальных парах слов. Именно такая вычислимая функция и будет точным аналогом универсальной функции из абстрактной теории.

Другими словами, мы допускаем «нестандартное» кодирование некоторых алгоритмов. Это не создаёт проблему, хотя при желании можно построить вычислимые биекции между кодами машин и всеми словами в алфавите $\{0,1,\#\}$, а также между кодами входных слов и всеми словами в алфавите $\{0,1,\#\}$.

16.8 Универсальная 3-ленточная машина для 1-ленточных машин

Мы построим 3-ленточную машину, которая вычисляет универсальную функцию. Поскольку любую 3-ленточную машину можно преобразовать в 1-ленточную, получим отсюда существование искомой 1-ленточной МТ.

Для вычисления универсальной функции 3-ленточная УМТ U моделирует работу произвольной МТ на произвольном входе. Идея моделирования достаточно проста. На одной ленте (лента машины) машина U хранит описание моделируемой МТ M, на второй (лента состояния) держит описание текущего состояния и текущего символа, на третьей (рабочая лента, она же входная и лента результата) поддерживает описание ленты моделируемой машины. УМТ начинает работу, приводя начальную конфигурацию к указанному виду. После этого работа машины M моделируется такт за тактом.

Опишем более подробно эти действия. Все они основаны на МТ, которые копируют или переносят часть одной ленты на другую, а также МТ, которые сравнивают слова на двух лентах.

Задача 16.7. Постройте (многоленточные) МТ, которые выполняют следующие действия.

(а) «Переход к указателю»: машина двигает головку до тех пор, пока головка не оказывается над заданным символом (указателем) #. В терминах конфигураций это означает, что конфигурацию $\dots q_0 x \# \dots$ машина должна переводить в конфигурацию $\dots x q_1 \# \dots$. Здесь q_0, q_1 — состояния МТ, а x — слово, которое не содержит указателя.

(Будет также использоваться и переход влево к указателю.)

(б) «Копирование»: машина копирует содержимое области одной ленты на другую ленту. В терминах конфигураций это означает, что машина переводит конфигурацию

$$(\dots q_0 \triangleleft u \triangleright \dots \ , \ \dots q_0 \# \dots)$$

в конфигурацию

$$(\dots q_1 \triangleleft u \triangleright \dots \ , \ \dots q_1 \# u \dots).$$

Здесь q_0, q_1 — состояния МТ, символы-ограничители $\triangleleft$, $\triangleright$ задают область, которую нужно скопировать, # — указатель, а слово u не содержит ни указателя, ни ограничителей.

(Конфигурация двухленточной машины — это пара слов.)

(в) «Перенос»: машина переносит содержимое области одной ленты на другую ленту. В терминах конфигураций это означает, что машина переводит конфигурацию

$$(\dots q_0 u \triangleright \dots \ , \ \dots q_0 \dots)$$

в конфигурацию

$$(\dots q_1 \triangleright \dots \ , \ \dots q_1 u \dots).$$

Здесь q_0, q_1 — состояния МТ, символы-ограничители $\triangleleft$, $\triangleright$ задают область, которую нужно скопировать, # — указатель, а слово u не содержит ограничителя $\triangleright$.

(г) «Сравнение слов»: машина сравнивает две области на двух лентах. В терминах конфигураций это означает, что машина переводит конфигурацию

$$(\dots q_0 \triangleleft u \triangleright \dots \ , \ \dots q_0 \triangleleft v \triangleright \dots)$$

в конфигурацию

$$(\dots q' \triangleleft u \triangleright \dots \ , \ \dots q' \triangleleft v \triangleright \dots),$$

где $q' = q_1$ если $u = v$, а в противном случае $q' = q_2$. Здесь q_0, q_1, q_2 — состояния МТ, символы-ограничители $\triangleleft$, $\triangleright$ задают сравниваемые области, а слова u, v не содержат ограничителей.

Кроме того универсальная машина должна уметь считать до пяти (строки таблицы переходов — это пятёрки). Мы не указываем явно расширенного множества состояний, которое позволяет это сделать, см. общие замечания об использовании «оперативной памяти» в МТ.

Универсальная машина получается последовательным соединением трёх машин: (1) подготовительной, (2) моделирующей такт работы, (3) финальной.

Подготовительная машина переносит с входной ленты на ленту описания МТ часть входа до символа $\odot$ (см. задачу 16.7в). Символ $\odot$ удаляется на первой ленте и головка сдвигается вправо. Если описание машины пусто, то подготовительная машина останавливается без запуска машины, моделирующей такт работы.

Далее подготовительная машина помещает на ленту состояния код начального состояния 10^k. Для этого она использует второй код в первой строке таблицы переходов и машину из задачи 16.7в. Однако скопированное состояние может и не быть начальным (мы не оговаривали порядок строк в таблице переходов). Поэтому подготовительная машина пишет на первую позицию кода 1, а на все последующие позиции вплоть до символа # она записывает 0, чтобы на второй ленте и впрямь появился код начального состояния[2], после чего переходит влево до символа # (задача 16.7а). На ленте описания МТ подготовительная машина переводит головку в крайнее левое положение (т.е. переходит влево по указателю #).

На том работа подготовительной машины завершается.

Корректность работы машины очевидна из сделанного описания за одним исключением. Подготовительная машина берётся определить результат работы *пустой машины* — те такой МТ, таблица переходов которой не определена ни для одной пары (символ, состояние). Из построения видно, что результат оказывается равным входу. Это соответствует работе пустой машины — та останавливается на первом же такте и не сдвигает головку, поэтому вычисляет тождественную функцию.

Машина, моделирующая такт работы. Подготовительная машина заканчивает работу в такой конфигурации: на первой ленте (описания МТ) и второй ленте (описание текущего состояния) головки находятся над самой левой непустой ячейкой

[2]Построение такой вспомогательной машины оставляем читателю в качестве упражнения.

(содержащей разделитель #), а на третьей ленте головка находится над разделителем перед кодом символа в текущей ячейке машины M.

Моделирование такта работы машины M, описание которой лежит на ленте описания МТ, будет начинаться и заканчиваться именно в таких конфигурациях.

Моделирование такта работы состоит в выполнении следующих шагов, которые, как и выше, используют МТ из задачи 16.7 и переходы между разделителями на заданное число позиций (итерация МТ из задачи 16.7а нужное количество раз, которое не превосходит 5):

(1) Перейти вправо до # по ленте состояния и скопировать на эту ленту код текущего символа.

(2) Найти строчку таблицы переходов, которая соответствует текущему состоянию и символу моделируемой машины. Для этого последовательно проверять пятёрки кодов на ленте описания МТ (это строчки таблицы переходов моделируемой машины M), для каждой пятёрки первые два кода сравнить с текущим символом и текущим состоянием соответственно (задача 16.7г применительно к рабочей ленте и ленте состояния).

   – В случае совпадения искомая строчка найдена, перейти влево на первый символ # этой строчки.
   – В случае несовпадения перейти вправо на первый символ # следующей строчки.

   Если совпадения не найдено, управление передаётся финальной машине. Иначе выполняются следующие действия:

(3) скопировать четвёртый код в пятёрке на ленту состояния;

(4) скопировать третий код в пятёрке на рабочую ленту;

(5) выполнить команду движения: переход влево или вправо до символа # на рабочей ленте; если сдвиг выходит за границы закодированного слова, т.е. машина встретила пустой символ машины U, то добавить на рабочую ленту код пустого символа аналогично тому, как это делает подготовительная машина с кодом начального состояния.

Чтобы убедиться в корректности такой машины (т.е. проверить, что конфигурация изменяется в соответствии с тактом работы машины M), достаточно сравнить формат описания МТ, данный выше с указанной последовательностью действий.

Финальная машина помечает текущую позицию, после чего ищет справа первый код пустого символа. Поскольку этот код 10^n, то сравнение символа с пустым выполняется без труда. Далее машина пишет пустой символ (машины U) и возвращается в помеченную исходную позицию и убирает метку.

Итак, мы построили 3-ленточную машину, вычисляющую универсальную функцию. Поскольку 3-ленточные машины моделируются на 1-ленточных, существует и 1-ленточная машина, вычисляющая универсальную функцию.

16.9 Соответствие между абстрактной теорией алгоритмов и МТ

Мы уже реализовали на машинах Тьюринга два самых важных свойства алгоритмов: доказали, что композиция вычислимых МТ функций вычислима и построили универсальную машину Тьюринга.

Но этого пока недостаточно, чтобы доказать неразрешимость следующей *проблемы остановки*: даны описание машины Тьюринга и её входа, нужно узнать, останавливается ли эта машина на этом входе.

Действительно, мы доказывали, что неразрешимо множество $H = \{n : U(n, n) \text{ определена}\}$. Сейчас под n нужно понимать слово в алфавите $\{0, 1, \#\}$. Заметьте, что построенная нами универсальная функция вполне возможно даёт какой-то результат и для слов, которые не являются описаниями МТ и их входов. Вдруг окажется так, что вся «трудность» диагонального множества H прячется именно в таких «паразитных» парах?

Задача 16.8. Какой результат даёт построенная выше УМТ на парах $\langle M\rangle \odot \langle x\rangle$, где алфавит M состоит из 10 символов, а слово x в алфавите из 5 символов?

Для преодоления этой трудности необходимо использовать эффективность выбранного нами способа кодирования МТ.

Задача 16.9. Докажите, что на машинах Тьюринга разрешимы следующие множества слов:

(а) слова вида $\langle M\rangle$, где M — машина Тьюринга;

(б) слова вида $\langle x\rangle$, где x — слово в некотором алфавите;

(в) слова вида $\langle M\rangle \odot \langle x\rangle$, где M — машина Тьюринга, x — входное слово для M.

Разрешимость множества слов X означает, что существует МТ, которая даёт результат 1 на словах из X и результат 0 на остальных словах.

Теорема 16.4. *Проблема остановки алгоритмически неразрешима.*

Доказательство. Используем результат предыдущей задачи. Подправим универсальную функцию: на паре (u, v) подправленная функция даёт результат $\langle M(x)\rangle$, если $u = \langle M\rangle$, $v = \langle x\rangle$, и не определена иначе. Вычислимость такой подправленной функции следует из разрешимости множества пар (описание МТ, описание входа МТ), при этом она остаётся универсальной.

Если бы проблема остановки МТ была разрешима, то было бы разрешимым и диагональное множество для такой подправленной функции: алгоритм разрешения на входе w проверяет, является ли $w \odot w$ описанием МТ и её входного слова; если да, то применяет алгоритм разрешения для проблемы остановки, а если нет, то говорит, что w не принадлежит диагональному множеству. □

Впрочем, неразрешимость проблемы остановки можно доказать и без высокой теории, прямым применением диагонального рассуждения.

Задача 16.10. Выведите напрямую из тезиса Чёрча–Тьюринга неразрешимость проблемы остановки (без использования существования УМТ или абстрактной теории алгоритмов).

Опишем и общий способ сопоставить абстрактную теорию алгоритмов и построенную нами универсальную функцию для машин Тьюринга. Для этого упорядочим слова в алфавите $\{0, 1, \#\}$ по правилу: более короткое слово меньше более длинного, а слова одинаковой длины упорядочиваются лексикографически. Этот порядок линейный и фундированный.

Задача 16.11. Постройте МТ, которая по слову в алфавите $\{0, 1, \#\}$ находит следующее слово в указанном порядке.

Используя задачи 16.9 и 16.11, легко построить вычислимые на МТ нумерации как МТ, так и входных слов для этих машин. Это позволяет ограничиться только значениями универсальной для МТ функции только на парах (описание МТ, описание входного слова). Действительно, на машинах Тьюринга вычислима такая универсальная функция от двух натуральных аргументов n и k с натуральными значениями: найдём n-ую машину M_n в порядке описаний МТ и k-е слово x_k в порядке описания входных слов машины M_n, значением функции будет номер результата $M_n(x_k)$ в порядке описания входных слов машины M_n.

Задание алгоритмов машинами Тьюринга — это главная нумерация. Действительно, пусть функция двух аргументов $V(n, x)$ вычислима МТ M, т.е. на входе вида $n \odot x$ машина M даёт результат $V(n, x)$. Как построить алгоритм, который по n находит описание МТ, вычисляющей функцию $f_n(x) = V(n, x)$? Одной из машин, вычисляющих f_n, является такая, которая ко входу x дописывает слева разделитель $\odot$, а потом первый аргумент n; далее она выполняет действия машины, вычисляющей $V(n, x)$. Описание такой машины легко строится по входу n: нужно описать машину, которая пишет разделитель, а потом слово n символ за символом в обратном порядке; потом добавить к ней фиксированное описание машины M.

Задача 16.12. Докажите существование МТ, реализующей указанное выше преобразование.

При изучении главных нумераций мы активно использовали биекцию между парами чисел и числами.

Задача 16.13. Докажите, что существует МТ, которая вычисляет биекцию $c\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$, где

$$c\colon (x, y) \mapsto \binom{x + y + 1}{2} + y.$$

Числа заданы в унарной записи (т.е. n записывается как слово 1^n).

Как определить перечислимые множества, если в качестве алгоритмов используются машины Тьюринга? Для этого нужно определить машину Тьюринга, которая порождает список элементов множества. Будем считать, что такая перечисляющая

машина имеет специальную ленту результата, куда записывает элементы списка через разделитель #. При этом перечисляющая машина сдвигает головку на ленте результата только вправо (такое ограничение легко выражается в терминах таблицы переходов).

Ещё мы использовали в абстрактной теории отладочную функцию и возможность запуска нескольких алгоритмов одновременно.

Задача 16.14. Докажите существование МТ, которая по входу $\langle M\rangle\#\langle x\rangle\#1^t$ проверяет, останавливается ли машина M на входе x за t шагов.

Параллельное исполнение нескольких МТ (заданного числа) легче всего смоделировать, воспользовавшись многоленточными машинами. Но возможно реализовать также и параллельное исполнение МТ на всех её входах, разбивая работу на стадии: первая стадия состоит в исполнении первого такта на первом входе, вторая — второго такта на первом входе и первого такта на втором входе и т.д.: на N-й стадии исполняется N-й такт на первом входе, $(N-1)$-й такт на втором входе, ..., первый такт на N-м входе.

Задача 16.15. Докажите существование (многоленточной) МТ, которая реализует указанный выше процесс.

Используя эти соответствия, можно перенести все доказанные в абстрактной теории утверждения на машины Тьюринга. Приведём несколько следствий:

Следствие 16.5 (теорема Райса)**.** *Для любого нетривиального свойства $\mathcal{A}$ функций алгоритмически неразрешима задача: по описанию МТ проверить, вычисляет ли эта МТ функцию, обладающую свойством $\mathcal{A}$.*

Следствие 16.6. *Неперечислимо множество описаний МТ, которые не останавливаются на любом входе.*

Следствие 16.7. *Существует МТ, которая на любом входе выдаёт в качестве результата своё описание.*