Московский государственный технический университет
имени Н.Э. Баумана

Факультет «Фундаментальные науки»
Кафедра «Математическое моделирование»

А.Н. Канатников

МАТЕМАТИЧЕСКАЯ ЛОГИКА И ТЕОРИЯ АЛГОРИТМОВ

Конспект лекций

Для студентов кафедры ИУ9

Москва
2010

10. МАШИНЫ ТЬЮРИНГА

10.1. Основные понятия

Следующий вариант строгого определения алгоритма — представление результата как результата работы некоего автоматического устройства — машины. В принципе машина — дискретно работающее устройство, которое в каждый момент времени находится в одном из внутренних состояний. Внутреннее состояние меняется за один шаг в зависимости от текущего состояния и входных данных. За один шаг машина формирует какие-то выходные данные.

Разумеется, результат действий абстрактной машины можно описать без „костылей“ — представления их как результат деятельности некоторого механического устройства. Есть объект, который описывает доступную в данный момент входную информацию, есть множество внутренних состояний. В этом контексте речь идет о некотором отображении, которое паре множеств ставит в соответствие новое множество — множество выходных данных.

Машина Тьюринга — простейший вариант описанной абстрактной машины. Есть целый ряд ее модификаций. Опишем одну из них. Зададимся некоторым непустым алфавитом $A = \{a_0, a_1, \ldots, a_n\}$, который назовем ***внешним алфавитом***. В этом алфавите буква a_0 играет особую роль. Введем еще ***внутренний алфавит*** $Q = \{q_0, q_1, q_2, \ldots, q_m\}$, определяющий ***множество внутренних состояний*** машины. Считаем, что внутренний алфавит имеет по крайней мере два элемента q_0 и q_1, играющих специальную роль. В объединенном алфавите дополнительно включим четыре спецсимвола $\to$, R, L, C составляем слова специального вида $q_i a_j \to q_l a_k M$, где $1 \leqslant i \leqslant m$, $0 \leqslant l \leqslant m$, $0 \leqslant j, k \leqslant n$, а M — один из трех символов R, L, C. Множество таких слов, имеющих разные левые части (два символа перед $\to$), назовем ***программой*** (заметим, что символ $\to$ на самом деле не нужен и используется лишь для наглядности).

В дальнейшем символ a_0 будем называть ***пустым символом***, символ q_0 — ***заключительным состоянием***, символ q_1 — ***начальным состоянием***.

Действие машины состоит в следующем. Есть лента, разделенная на конечное число ячеек. Лента потенциально бесконечная, т.е. в любой момент к ней можно добавлять ячейки и слева (в начало) и справа (в конец)*. Каждая ячейка заполнена одним символом внешнего алфавита. Добавляемые ячейки заполняются пустым символом a_0. У машины есть ***читающая головка***, которая обозревает ленту. В каждый момент времени головка обозревает одну ячейку с некоторым символом a_j и находится в одном из внутренних состояний q_i. За один шаг работы машина изменяет свое внутреннее состояние, меняет обозреваемый головкой символ на ленте и может на одну позицию влево или вправо сместить положение читающей головки. Это происходит в соответствии с командой $q_i a_j \to q_l a_k M$, которая вызывает замену текущего символа a_j символом a_k, смену внутреннего состояния с q_i на q_l. При этом, если $M = R$, то читающая головка смещается вправо, если $M = L$, то влево, если же $M = C$, то остается на месте.

Шаги работы машины повторяются до тех пор, пока не произойдет одно из двух событий: 1) для текущего состояния и обозреваемого символа нет команды; 2) текущая команда привела к заключительному внутреннему состоянию q_0. В этом случае машина останавливается, а слово, оказавшееся на ленте, считается результатом работы машины.

В действительности представления о ленте и читающей головке, хотя и являются общеупотребительными, нужны лишь для наглядности. С математической точки зрения эти понятия

*Можно сказать и по-другому: лента бесконечная в обе стороны, но количество ячеек, заполненных непустыми символами, конечно.

ничтожные (как выражаются юристы). Вместо трех понятий „текущее состояние“, „обозреваемый символ“, „лента“ введем одно понятие: ***конфигурация*** (или машинное слово). Это понятие обозначает слово вида Xq_la_jY, где $X, Y \in A^*$, т.е. слово во внешнем алфавите A, в которое добавлена одна буква из внутреннего алфавита. Буква внутреннего алфавита отмечает текущее состояние машины Тьюринга, следующий за ней символ будет обозреваемым. Так, в конфигурации $a_3a_2q_1a_2a_1$ зафиксировано текущее состояние q_1 (начальное) и обозреваемый символ a_2. Машина Тьюринга реализует преобразование одной конфигурации в другую.

Дадим формальное описание действий, совершаемых машиной Тьюринга за один такт. Предполагаем, что текущая конфигурация имеет вид Xq_ia_jY.

• если в программе машины есть команда $q_ia_j \to q_ka_lR$ и $Y \neq \varnothing$, то новая конфигурация Xa_lq_kY;

• если в программе машины есть команда $q_ia_j \to q_ka_lR$ и $Y = \varnothing$, то новая конфигурация $Xa_lq_ka_0$;

• если в программе машины есть команда $q_ia_j \to q_ka_lL$ и $X = X_1a_m \neq \varnothing$, то новая конфигурация $X_1q_ka_ma_lY$;

• если в программе машины есть команда $q_ia_j \to q_ka_lL$ и $X = \varnothing$, то новая конфигурация $q_ka_0a_lY$;

• если в программе машины есть команда $q_ia_j \to q_ka_lC$, то новая конфигурация Xq_ka_lY;

Если конфигурация M' получена из конфигурации M за один такт работы машины Тьюринга, то будем говорить, что машина Тьюринга переводит M в M'. Машина Тьюринга преобразует конфигурацию M в конфигурацию $\tilde{M}$, если существует такая последовательность конфигураций $M_0 = M$, M_1, …, $M_n = \tilde{M}$, что конфигурация M_i, $i = \overline{0, n-1}$, машиной Тьюринга переводится в конфигурацию M_{i+1}. ***Вычислением*** машины Тьюринга назовем такое ее преобразование M в $\tilde{M}$, что конфигурация $\tilde{M}$ является заключительной (терминальной), т.е. либо нет команды, соответствующей ее левой части, либо эта конфигурация соответствует заключительному состоянию.

Существуют разные модификации машины Тьюринга, в основном различающиеся в трех аспектах.

Во-первых, операции сдвига читающей головки могут оформляться как самостоятельные, т.е. при сдвиге головки обозреваемый символ не изменяется (это так называемый вариант Поста). В этом случае команды машины имеют вид $q_ia_j \to q_lb$, где $b \in A \cup \{L, R\}$. Если дана, например, команда $q_ia_j \to q_lR$, то машина изменяет свое внутреннее состояние с q_i на q_l и сдвигает вправо читающую головку, а обозреваемый символ a_j не изменяется.

Вариант Поста машины Тьюринга можно считать частным случаем рассматриваемого варианта (варианта Клини), поскольку всегда команды вида $q_ia_j \to q_lL$ и $q_ia_j \to q_lR$ можно трансформировать в команды $q_ia_j \to q_la_jL$ и $q_ia_j \to q_la_jR$, а команды вида $q_ia_j \to q_la_k$ — в команды $q_ia_j \to q_la_kC$. В то же время можно произвольную программу трансформировать так, что операции изменения символа на ленте и смещения читающей головки будут выполняться раздельно. Для этого достаточно для каждого состояния q_i ввести два новых состояния q_i^p — сдвиг влево и q_i^s — сдвиг вправо. Команды для этих состояний должны иметь вид $q_i^p\xi \to q_i\xi L$ и $q_i^s\xi \to q_i\xi R$ (здесь ξ — любой символ внешнего алфавита). Затем команду с левым сдвигом читающей головки вида $q_ia_j \to q_ma_kL$ заменяем командой $q_ia_j \to q_m^pa_k$ и аналогично изменяем команды с правым сдвигом читающей головки.

Во-вторых, различия проявляются в условиях завершения. В описанном варианте есть два сценария останова машины Тьюринга: либо по отсутствию команды для текущего положения машины (т.е. комбинации внутреннего состояния и обозреваемого символа), либо по наступающему заключительному внутреннему состоянию. Эта ситуация аналогична нормальным алгорифмам. В действительности нет необходимости объявлять какое-либо состояние заключительным: заключительным является всякое состояние, которое не встречается в левой части команд в программе машины Тьюринга. Правда, в описанном варианте могут быть условно

заключительные состояния, которые приводят к останову лишь для некоторых обозреваемых символов.

Программу машины Тьюринга легко модифицировать так, что останов по отсутствию команды не будет происходить. Достаточно из программы удалить все команды для заключительного состояния q_0 (они все равно никогда не выполняются) и для каждого положения $q_i a_j$, для которого нет команды, добавить команду $q_i a_j \to q_0 a_j$. В результате программа будет содержать ровно $m(n+1)$ (количество незаключительных состояний на количество букв внешнего алфавита) команд, описывающих все незаключительные положения. Такую процедуру будем называть ***замыканием машины Тьюринга***.

В-третьих, машины Тьюринга различаются входными и выходными потоками. Так, рабочая лента может быть полуограниченной, т.е. неограниченное перемещение влево оказывается недоступно. Кроме того, могут использоваться несколько лент. Подробнее об этом будет сказано далее.

Далее в теоретических рассуждениях будем в основном предполагать, что машина Тьюринга замкнута, в то время как в конкретных примерах будем использовать незамкнутые программы, удаляя тривиальные команды типа $q_i a_j \to q_0 a_j m$, которые ничего не делают на ленте, а лишь приводят к останову машины Тьюринга.

С машиной Тьюринга можно связать словарную функцию следующим образом. Исходное слово X в алфавите A превращается в конфигурацию M_X приписыванием символа q_1 начального состояния к слову слева (т.е. фиксируется начальное состояние q_1 и читающая головка устанавливается на первый символ слова). Машина Тьюринга начинает работу с этой конфигурации. Если она останавливается на некоторой конфигурации M_T (по отсутствию команды или по заключительному состоянию — неважно), то слово, полученное из M_T удалением символа состояния, а также всех пустых символов в начале и конце слова считаем значением словарной функции на слове X. Если машина Тьюринга работает бесконечно (зацикливается), то считаем, что словарная функция не определена на слове X.

Пример 10.1. Рассмотрим алфавит $A = \{a,\ b,\ c\}$ и машину Тьюринга с программой

$$\begin{cases} q_1 b \to q_1 c R, \\ q_1 c \to q_1 b R, \\ q_1 a \to q_0 a. \end{cases}$$

Машина имеет два состояния q_0, q_1 (заключительное и начальное). Как сказано выше, начальная конфигурация машины Тьюринга $q_1 X$, где X — исходное слово. В процессе работы машины делает взаимные замены символов b и c, пока не встретит символ a (он играет роль пустого символа). Так, слово bab машина преобразует в слово cab, слово $bcbc$ — в слово $cbcb$ (останов произойдет, когда будет достигнут конец слова и для движения вправо будет добавлен пустой символ; этот символ в окончательном результате не учитывается).

Выделяют также в некотором смысле регулярные сценарии работы машины Тьюринга (***правильные вычисления***), при которых не происходит добавления пустых символов к слову справа и/или слева.

С помощью машины Тьюринга можно вычислять числовые функции $f(x_1, x_2, \ldots, x_n)$, $x_i \in \mathbb{N}$. Для этого выбираем машину Тьюринга с внешним алфавитом $A = \{0,\ 1\}$, в котором 0 играет роль пустого символа. Аргументы функции записываем в виде 0-системы, т.е. в виде $X = 01^{x_1}01^{x_2}0 \ldots 01^{x_n}0$. Машина Тьюринга с алфавитом $\{0,\ 1\}$ вычисляет функцию f, если соответствующая ей словарная функция преобразует слово X в слово 01^y0, где $y = f(x_1, x_2, \ldots, x_n)$, и не определена на слове X, если функция f не определена на сочетании аргументов x_1, x_2, …, x_n. Если функция $f(x_1, \ldots, x_n)$ может быть вычислена с помощью некоторой машины Тьюринга, ее называют ***вычислимой по Тьюрингу***.

Пример 10.2. Машина Тьюринга с алфавитом 01 и программой $q_1 0 \to q_0 1$, $q_1 1 \to q_1 1 L$ вычисляет функцию $f(x) = x + 1$ (инкремент). Отметим, что алгоритм использует смещение

читающей головки за левую границу слова. Для правильного вычисления инкремента, при котором не используется смещение за левую границу (или еще более жестко — и за правую границу), а заключительное положение читающей головки — первый символ слова, можно использовать машину Тьюринга с программой

$$\begin{cases} q_1 0 \to q_2 0 R, \\ q_1 1 \to q_1 1 \quad \text{(Неверное слово!)}, \\ q_2 1 \to q_2 1 R, \\ q_2 0 \to q_3 1, \\ q_3 1 \to q_3 1 L, \\ q_3 0 \to q_0 0. \end{cases}$$

Эта программа смещает читающую головку на конец слова, где завершающий нуль меняется на единицу, а затем головка смещается к началу слова, определяемому символом 0.

10.2. Сочетания машин Тьюринга

Как и нормальные алгорифмы, машины Тьюринга можно сочетать определенным образом. Базовой операцией является композиция машин Тьюринга.

Пусть T_1 и T_2 — две машины Тьюринга с внутренними алфавитами $Q_1 = \{q_0^1,\ q_1^1,\ \ldots,\ a_{m_1}^1\}$ и $Q_2 = \{q_0^2,\ q_1^2,\ \ldots,\ a_{m_2}^2\}$, имеющие одинаковый внешний алфавит $A = \{a_0,\ a_1,\ \ldots,\ a_n\}$. Пусть этим машинам соответствуют словарные функции f_1 и f_2. Существует машина Тьюринга с внешним алфавитом A, словарная функция f которой удовлетворяет условию $f(X) \simeq f_2(f_1(X))$. Такую машину можно построить, используя стандартную процедуру объединения машин T_1 и T_2. Выберем какое-либо расширение $\{q_{m_1+1}^1,\ \ldots,\ q_{m_1+m_2}^1\}$ внутреннего алфавита Q_1 машины T_1. В программе P_1 машины T_1 заменим все вхождения символа q_0^1 символом $q_{m_1+1}^1$, а в программе P_2 машины T_2 сделаем следующие замены: $q_0^2 \to q_0^1$, $q_j^2 \to q_{m_1+j}^1$, $j = \overline{1,\ m_2}$. После этого измененные программы объединим. Получим программу машины T, которую и назовем ***композицией машин Тьюринга*** T_1 и T_2.

Замечание 10.1. Поскольку ключевой целью машины Тьюринга является вычисление некоторой словарной функции, следовало бы ввести отношение эквивалентности между машинами, считая эквивалентными те, которые вычисляют одну и ту же словарную функцию. При этом возможны два уровня эквивалентности, как и в нормальных алгорифмах. Мы этого делать не будем, отметив однако, что композиция двух машин может осуществляться разными машинами.

Довольно трудно устроить соединение машин Тьюринга, так как механизм их работы заметно другой. В первую очередь трудно реализовать вставку символа в слово (нормальный алгорифм это реализует легко). Надо сдвигать все символы вправо, а потом на освободившееся место записать нужный символ.

В то же время разветвление машин сделать гораздо проще: никаких специальных управляющих алгоритмов не нужно, для выбора разных действий достаточно использовать разные внутренние состояния. Пусть даны три (замкнутые) машины Тьюринга T_1, T_2, T_3 с общим внешним алфавитом A и внутренними алфавитами $Q_s = \{q_0^s,\ q_1^s,\ \ldots,\ q_{m_s}^s\}$, $s = 1, 2, 3$ (полагаем, что три алфавита не имеют общих символов). Выделим в машине T_1 два внутренних состояния q_i^1 и q_j^1, $i,\ j \neq 0$. Объединим программы трех машин, предварительно выполнив в них следующие преобразования. В программе машины T_1 удаляем все команды, у которых в левой части указано состояние q_i^1 или q_j^1. В машине T_2 меняем внутренние состояния q_0^2 на q_0^1, q_1^2 на q_i^1. Аналогично в машине T_3 меняем внутренние состояния q_0^3 на q_0^1, q_1^3 на q_j^1. После этого объединяем три программы. Результирующая машина будет работать так. Стартовое

состояние q_1^1 начинает работу машины T_1. Если будет достигнуто состояние q_0^1, машина остановится, так что ни T_2, ни T_3 не будут использованы. Если будет достигнуто состояние q_i^1, начнет работу машина T_2, для которой это состояние стало стартовым. Завершение работы машины T_2 приведет к состоянию q_0^1, являющемуся заключительным. Если будет достигнуто состояние q_j^1, то далее будет работать машина T_3, которая остановит работу в состоянии q_0^1.

По этой же схеме легко устроить разветвление на большее число вариантов.

Замечание 10.2. И в композиции, и в разветвлении предполагается, что рассматриваемые алгоритмы заканчивают работу в стандартном положении, когда читающая головка обозревает первый символ слова. Если это условие не выполняется, то при передаче управления от одной машины к другой нужно вставлять специальную машину, которая не меняет данных, а лишь передвигает читающую головку в нужное положение.

Важное обстоятельство состоит в том, что идентифицировать в процессе работы стартовый символ не так-то просто. Если слово не содержит пустых символов, то это можно сделать движением влево до пустого символа, правда, это приведет к смещению за левую границу слова. Еще один вариант: использовать алфавит двойников. Тогда, заменив в самом начале первый символ слова его двойником, мы легко сможем идентифицировать его в дальнейшем.

Как устроить повторение машины Тьюринга по типу повторения нормальных алгорифмов? Например, так. Рассмотрим в качестве алгоритма T_2 специальный алгоритм, который не изменяет слова на ленте, а лишь формирует состояние q_0, которое возникает при обозревании первого символа слова. Изменим состояние этой машины с q_0 на q_1. Тогда при достижении состояния q_i запускается машина T_2, которая организует повторный запуск машины T_1.

10.3. Эквивалентность машин Тьюринга и нормальных алгорифмов

Форма записи уже подсказывает, что программу машины Тьюринга легко трасформировать в схему нормального алгорифма. Действительно, всего пять сценариев одного такта машины Тьюринга: по два со смещением читающей головки и один без смещения. Выберем в качестве алфавита для нормального алгорифма внешний алфавит машины Тьюринга, а символы внутреннего состояния будем рассматривать как специальные символы (считаем, что машина Тьюринга замкнута). Заменим команду вида $q_i a_j \to q_l a_k L$ двумя подстановками $\xi q_i a_j \to q_l \xi a_k$, $q_i a_j \to q_l a_0 a_k$, команду вида $q_i a_j \to q_l a_k R$ — подстановками $q_i a_j \xi \to a_k q_l \xi$, $q_i a_j \to a_k q_l a_0$, а команду вида $q_i a_j \to q_l a_k C$ — подстановкой $q_i a_j \to q_l a_k$. В конец записываем подстановку $\Lambda \to q_1$, осуществляющую запуск нормального алгорифма. Перед командой запуска поставим терминальную подстановку $q_0 \to\cdot\, q_0$. Полученный нормальный алгорифм выполнит те же преобразования, что и машина Тьюринга, поскольку в текущем слове только один специальный символ, причем этот символ не является в слове последним. Стартовая подстановка в конце схемы формирует начальную конфигурацию. Следовательно, в результате работы сконструированного нормального алгорифма мы получим конечную конфигурацию машины Тьюринга. Чтобы получить требуемый результат, необходимо удалить пустые символы в начале и конце слова, а также символ q_0 конечного состояния. Для этого достаточно взять композицию построенного нормального алгорифма с алгоритмом, например, с такой схемой:

$$\begin{cases} \xi q_0 \to q_0 \xi, \\ q_0 a_0 \to q_0, \\ q_0 \to r, \\ r\xi \to \xi r, \\ a_0 r \to r, \\ r \to\cdot\, \Lambda. \end{cases}$$

Из приведенных рассуждений получаем следующий вывод.

Теорема 10.1. Любая словарная функция, вычислимая по Тьюрингу, вычислима и по Маркову. #

Доказанная теорема по существу означает, что возможности нормальных алгорифмов по крайней мере не хуже возможностей машин Тьюринга. Дополнительная мощь нормальных алгорифмов связана с нелокальным характером преобразования слова. Чтобы эту энергию ввести в управляемое русло, мы используем расширение алфавита — так называемые спецсимволы. Эти символы сродни символам, обозначающим внутренние состояния машины Тьюринга. И, в общем-то, ясно, как нормальный алгоритм с такими спецсимволами трансформировать в программу машины Тьюринга. Однако, во-первых, не все нормальные алгорифмы используют спецсимволы, а во-вторых, нормальный алгорифм может использовать одновременно несколько спецсимволов. В этом случае трансформировать нормальный алгоритм в машину Тьюринга сложнее. Вопрос, всегда ли такая трансформация возможна?

Теорема 10.2. Любая словарная функция, вычислимая по Маркову, вычислима и по Тьюрингу.

◀ Сумасшедшая идея — преобразовать в машину Тьюринга универсальный нормальный алгорифм. Но, во-первых, в универсальном алгорифме одновременно работают несколько спецсимволов, которые непосредственно интерпретировать как состояния машины не удастся, а во-вторых тогда надо составить еще машину, которая приписывает к слову изображение нормального алгорифма. Не факт, что это проще.

Можно сделать так. В качестве алфавита конструируемой машины Тьюринга рассматриваем алфавит нормального алгорифма с одним дополнительным символом, который будет играть роль пустого символа. Для каждой подстановки $B_i \to * C_i$ составляем машину Тьюринга T_i, которая ее реализует. Далее строим каскадное сочетание этих машин согласно схеме на рис. 10.1.

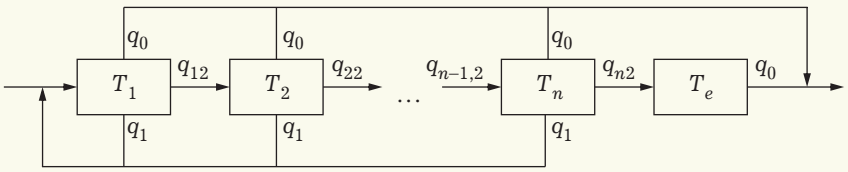

Рис. 10.1

Все машины T_i, $i = \overline{1, n}$, используют два общих состояния: q_0 — завершение работы всего комплекса (его формируют машины, реализующие терминальные подстановки, при успешной реализации такой подстановки и заключительная машина при неуспешной реализации); q_1 — повторение работы всего комплекса (его реализуют машины нетерминальных подстановок, причем каждая машина, завершая работу в этом состоянии, обеспечивает положение головки в начале слова). Остальные состояния машин T_i должны быть разными, и мы их будем обозначать q_{i2}, q_{i3} и т.д. Состояние q_{i2} должно быть заключительным в случае нереализованной подстановки, состояние q_{i3} начальным, причем $q_{i2} = q_{i+1,3}$, т.е. третье заключительное состояние каждой машины является начальным для следующей. При такой компоновке объединение всех машин даст машину Тьюринга, которая реализует схему нормального алгорифма.

Машину Тьюринга T_i, реализующую подстановку $B_i \to C_i$, можно реализовать как разветвление, построенное из трех составляющих машин:

• машина T_{i1} ищет первое вхождение подстроки B_i; если вхождение найдено, работа завершается в состоянии q_{iz} с положением читающей головки перед найденным вхождением; если вхождение не найдено, работа завершается в состоянии q_{i2} с положением читающей головки перед словом;

• машина T_{i2}, начиная работу в состоянии q_{iz}, заменяет найденное вхождение слова B_i словом C_i и заканчивает работу в состоянии q_1 (или q_0 в случае терминальной подстановки.

Пусть $B_i = b_1 b_2 \dots b_r$, $C_i = c_1 c_2 \dots c_s$. Машину T_{i1}, учитывая конкретный характер подстановки можно реализовать следующим образом:

$$\begin{cases} q_{i3}a_0 \to q_{i,r+5}a_0 L \\ q_{i3}b_1 \to q_{i4}uR, \\ q_{i3}\xi \to q_{i3}\xi R, \quad \xi \in A \setminus \{a_0,\ b_1\}, \\ q_{i4}b_2 \to q_{i5}b_2 R, \\ q_{i4}\xi \to q_{i,r+4}\xi, \quad \xi \in A \setminus \{b_2\}, \\ \dots\dots\dots\dots\dots\dots \\ q_{i,r+2}b_r \to q_{i,r+3}b_r R, \\ q_{i,r+2}\xi \to q_{i,r+4}\xi, \quad \xi \in A \setminus \{b_r\}, \\ q_{i,r+3}\xi \to q_{i,r+3}\xi L, \\ q_{i,r+3}u \to q_{iz}b_1 \quad \text{(успешное завершение)}, \\ q_{i,r+4}\xi \to q_{i,r+4}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,r+4}u \to q_{i3}b_1 R, \\ q_{i,r+5}a_0 \to q_{i2}a_0 R, \\ q_{i,r+5}\xi \to q_{i2}\xi R, \quad \xi \in A \setminus \{a_0\} \quad \text{(неуспешное завершение)} \end{cases}$$

Этот алгоритм завершает работу в состоянии q_{iz}, если сравнение удачное, причем положение читающей головки перед найденным фрагментом. Если вхождение неудачное, то завершаем работу в состоянии q_{i2}, причем положение читающей головки в начале данного слова.

Машина T_{i2} имеет три варианта конструкции в зависимости от соотношения длин r и s слов B_i и C_i. Если $r = s$, то машина T_{i2} реализуется без каких-либо проблем:

$$\begin{cases} q_{iz}b_1 \to q_{i,r+6}c_1 R, \\ q_{i,r+6}b_2 \to q_{i,r+7}c_2 R, \\ \dots\dots\dots\dots\dots\dots \\ q_{i,2r+4}b_r \to q_{i,2r+5}c_r R, \\ q_{i,2r+5}\xi \to q_{i,2r+5}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+5}a_0 \to q_1 a_0 R. \end{cases}$$

Эта машина завершает работу в состоянии q_1 и с положением читающей головки в начале слова.

Если $r > s$, то машину T_{i2} можно реализовать как композицию двух: первая машина T_{i21} заменяет B на C, заполняя лишние $r - s$ позиций пустым символом; вторая машина T_{i22} перемещает пустые символы в конец слова. Запишем машину T_{i21}:

$$\begin{cases} q_{iz}b_1 \to q_{i,r+6}c_1 R, \\ q_{i,r+6}b_2 \to q_{i,r+7}c_2 R, \\ \dots\dots\dots\dots\dots\dots \\ q_{i,r+s+4}b_s \to q_{i,r+s+5}c_s R, \\ q_{i,r+s+5}b_{s+1} \to q_{i,r+s+6}a_0 R, \\ \dots\dots\dots\dots\dots\dots \\ q_{i,2r+4}b_r \to q_{i,2r+5}a_0 R, \end{cases}$$

Машина завершает работу в состоянии $q_{i,2r+5}$ и с положением читающей головки вслед за полем из пустых символов. Машину T_{i22} можно реализовать следующим образом:

$$\begin{cases} q_{i,2r+5}\xi \to q_{i,\xi}uL, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,\xi}a_0 \to q_{i,\xi}a_0L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,\xi}\eta \to q_{i,\xi 1}\eta R, \quad \xi,\, \eta \in A \setminus \{a_0\}, \\ q_{i,\xi 1}a_0 \to q_{i,2r+6}\xi R, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+6}a_0 \to q_{i,2r+6}a_0R, \\ q_{i,2r+6}u \to q_{i,2r+5}a_0R, \\ q_{i,2r+5}a_0 \to q_{i,2r+7}a_0L, \\ q_{i,2r+7}a_0 \to q_{i,2r+7}a_0L, \\ q_{i,2r+7}\xi \to q_{i,2r+8}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+8}\xi \to q_{i,2r+8}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+8}a_0 \to q_1a_0R. \end{cases}$$

Эта машина завершает работу в состоянии q_1 с положением головки в начале слова.

Если $r < s$ то для вставки символов c_{r+1}, c_{r+2} надо раздвигать слово, освобождая место для вставляемого символа. Эту операцию лучше выполнить, модифицировав машину T_{i2} для случая $r = s$. Формируем машину T_{i21} следующим образом:

$$\begin{cases} q_{iz}b_1 \to q_{i,r+6}c_1R, \\ q_{i,r+6}b_2 \to q_{i,r+7}c_2R, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ q_{i,2r+4}b_r \to q_{i,2r+5}c_rR. \end{cases}$$

Теперь раздвигаем строку после символа c_r и вставляем туда символ c_{r+1}:

$$\begin{cases} q_{i,2r+5}\xi \to q_{i,\xi 2}a_0R, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,\xi 2}\eta \to q_{i,\eta 2}\xi R, \quad \xi,\, \eta \in A \setminus \{a_0\}, \\ q_{i,\xi 2}a_0 \to q_{i,2r+6}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+6}\xi \to q_{i,2r+6}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+6}a_0 \to q_{i,2r+7}c_{r+1}R. \end{cases}$$

Этот фрагмент повторяем для символов $c_{r+1}, \ldots c_s$:

$$\begin{cases} q_{i,2r+7}\xi \to q_{i,\xi 2}a_0R, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,\xi 2}\eta \to q_{i,\eta 2}\xi R, \quad \xi,\, \eta \in A \setminus \{a_0\}, \\ q_{i,\xi 2}a_0 \to q_{i,2r+8}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+8}\xi \to q_{i,2r+8}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2r+8}a_0 \to q_{i,2r+9}c_{r+2}R, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ q_{i,2s+3}\xi \to q_{i,\xi 2}a_0R, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,\xi 2}\eta \to q_{i,\eta 2}\xi R, \quad \xi,\, \eta \in A \setminus \{a_0\}, \\ q_{i,\xi 2}a_0 \to q_{i,2s+4}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2s+4}\xi \to q_{i,2s+4}\xi L, \quad \xi \in A \setminus \{a_0\}, \\ q_{i,2s+4}a_0 \to q_{i,2s+5}c_s. \end{cases}$$

Остается установить читающую головку перед словом и завершить работу в состоянии q_1 или q_0:

$$\begin{cases} q_{i,2s+5}a_0 \to q_1 a_0 R, \\ q_{i,2s+5}\xi \to q_{i,2s+5}\xi L, \quad \xi \in A \setminus \{a_0\}. \end{cases}$$

Машина T_e представляет собой „заглушку“: она просто переводит состояние q_{n2} в состояние q_0.

В совокупности рассмотренный комплекс машин T_i реализует нормальный алгорифм с заданной схемой. ►

10.4. Обобщения машин Тьюринга

Доказанная эквивалентность машин Тьюринга нормальным алгорифмам подчеркивает ту мысль, что машины Тьюринга позволяют реализовать любой мыслимый алгоритм или, по-другому, вычислить любую эффективно вычислимую словарную функцию. Однако на практике, как и в языках программирования, для решения конкретных задач удобнее использовать не один-единственный универсальный язык программирования, а целое множество таких языков, из которого можно выбрать наиболее подходящий. Так и в теории алгоритмов: предложено много различных вариаций абстрактных машин, да и иных концепций, которые предназначены для формализации понятия алгоритма.

Уже отмечалось, что простейшая машина Тьюринга имеет несколько вариаций. Однако эти вариации схожи по конструкции и виду элементарных операций. Есть, однако, конструкции абстрактных машин, которые получены усложнением (или, так сказать, усилением) первоначальной машины Тьюринга. Рассмотрим некоторые варианты.

Одноленточная машина с входом и выходом — это набор $\{I,\ O,\ W,\ Q,\ P\}$ из пяти множеств, в котором I — входной алфавит, O — выходной алфавит, W — рабочий алфавит, Q — множество состояний, P — программа. Команды программы имеют вид $q_i x_j w_k \to q_r w_s y_t M$, где:

$q_i \in Q$ — исходное состояние на данном такте;
$x_j \in I$ — входной символ (поступает с входной ленты);
$w_k \in W$ — обозреваемый символ рабочей ленты;
$q_r \in Q$ — результирующее состояние на данном такте;
$w_s \in W$ — символ, заменяющий обозреваемый символ рабочей ленты;
$y_t \in O$ — выходной символ (выводится на выходную ленту);
$M \in \{L,\ R,\ C\}$ — указатель движения головки на рабочей ленте.

На каждом такте с входной ленты поступает очередной символ x_j. Машина в соответствии с этим символом, внутренним состоянием q_i и обозреваемым символом w_k на рабочей ленте выполняет одновременно три действия: меняет внутреннее состояние на q_r, меняет обозреваемый символ на рабочей ленте на символ w_s и выводит на выходную ленту символ y_t. В начальном состоянии машина находится во внутреннем состоянии q_1, рабочая лента пуста. Завершение работы машины можно устроить по-разному. Например, как и в случае машины без входа и выхода, останов происходит при достижении внутреннего состояния q_0 (завершающего). Тогда исходное слово — последовательность использованных входных символов, результирующее слово — последовательность выходных символов. Как входные символы, так и выходные могут быть пустыми и в итоге не учитываются (нужны, чтобы сделать паузу на входе и выходе). Входная и выходная ленты бесконечны справа. Другой вариант, когда внутреннее завершающее состояние отсутствует, а машина останавливается, когда на выходной ленте появляется один из некоторого набора специальных символов. В этом случае нужен некий символ #, фиксирующий конец входного слова. Когда входное слово завершено, на вход подается символ # и с этого момента появление завершающего символа на выходе останавливает машину.

Многоголовочная машина Тьюринга имеет одну бесконечную двустороннюю ленту и некоторое количество k головок, обозревающих эту ленту. Положения головок независимы, команды такой машины имеют вид

$$q_i a_{j_1} a_{j_2} \dots a_{j_k} \to q_r a_{s_1} a_{s_2} \dots a_{s_k} M_1 M_2 \dots M_k. \tag{10.1}$$

При внутреннем состоянии q_i и обозреваемых символах a_{j_l}, $l = \overline{1,\ k}$, она переходит в состояние q_r, каждый l-й обозреваемый символ заменяется на a_{s_l}, а l-я головка смещается согласно символу $M_l \in \{L,\ R,\ C\}$. Независимое расположение головок на ленте может привести к ситуации, когда несколько головок обозревают один символ. В этом случае действует правило приоритета: за изменение символа отвечает головка с наименьшим номером (можем считать, что каждая головка сделает свое дело, но первой пишет головка с наибольшим номером). Останов такой машины может происходить либо по отсутствию команды, либо по заключительному состоянию (что, в общем-то, одно и то же).

Из этих монстров наиболее распространены ***многоленточные машины Тьюринга***, которые используют в задачах оценки сложности алгоритмов. Так называют абстрактную машину, имеющую несколько лент. На каждой ленте имеется читающая головка, все головки перемещаются независимо. Команды машины Тьюринга имеют тот же вид (10.1), но теперь головки находятся на разных лентах. Конфигурацией такой машины является набор слов $X_l q_i Y_l$, $l = \overline{1,\ k}$, где X_l, Y_l, — слова во внешнем алфавите, одно из которых не пусто. Начальная конфигурация состоит из слов вида $q_1 Y_l$ (т.е. $X_l = \Lambda$). Машина завершает работу, как и одноленточная машина, по отсутствию команды или по заключительному состоянию.

Многоленточные машины удобны для реализации словарных функций нескольких переменных. Каждый аргумент записывается на своей ленте. Одна лента выделена под результат. Некоторые ленты могут использоваться для промежуточных вычислений.

Обобщения машины Тьюринга потенциально имеют больше возможностей, чем стандартная машина Тьюринга. Так одноленточная машина может быть реализована на многоленточной, например, так: на одной ленте происходят вычисления, а остальные не изменяются. Чтобы обеспечить соглашение о расположении исходного слова и результата на разных лентах, достаточно реализовать предварительное копирование исходного слова на результирующую ленту. Однако в действительности расширение возможностей не приводит к расширению множества вычислимых словарных функций. Можно придумать разные варианты эмуляции многоленточной машины Тьюринга на одноленточной. Один из них такой. Исходные слова на лентах записываем как γ-систему, добавляя в слова специальный символ μ, указывающий положение головки на этом слове. Одноленточная машина пробегает слово, выбирая обозреваемые символы, которые идут вслед за μ, и формируя соответствующее внутреннее состояние. Затем обратным ходом машина реализует требуемые изменения. Разумеется, такая эмуляция требует большой внутренней памяти (множества внутренних состояний) и большой длины программы. Ясно, что в определенных ситуациях при решении задач на существование алгоритмов многоленточные машины использовать проще.

ОГЛАВЛЕНИЕ