Глава 6

МАШИНЫ ТЬЮРИНГА

§ 6.1. Неформальное и формальное описания

В этой главе мы рассмотрим еще один тип распознающих устройств — *машины Тьюринга*. Это абстрактное устройство было предложено в качестве математической модели для описания процедур. Так как наше интуитивное понятие процедуры как конечной последовательности инструкций, которые могут выполняться механически, не является математически точным, мы не можем доказать формально, что оно эквивалентно точному понятию машины Тьюринга. Однако из определения этого устройства будет очевидно, что любое вычисление, которое может быть описано посредством машины Тьюринга, может быть выполнено механически. Также может быть показано, что любое вычисление, которое может быть выполнено на современной вычислительной машине, может быть описано посредством машины Тьюринга. Таким образом, если кто-нибудь когда-нибудь нашел бы процедуру, которая соответствует интуитивным понятиям, но не поддается описанию посредством машины Тьюринга, то она была бы необычной природы, так как не могла бы быть запрограммирована для любой существующей вычислительной машины. Было предложено много других формализаций процедуры, и было показано, что все они эквивалентны формализации машины Тьюринга. Это поддерживает нашу уверенность в том, что машина Тьюринга имеет достаточную общность, чтобы охватить интуитивное понятие процедуры.

А. Чёрчем была высказана гипотеза, что любой процесс, который естественным образом мог бы быть назван процедурой, реализуем машиной Тьюринга. Впоследствии вычислимость при помощи машины Тьюринга стала признанным определением процедуры. Мы примем гипотезу Чёрча и просто подставим формальное определение машины Тьюринга вместо интуитивного понятия процедуры.

В литературе определение машины Тьюринга давалось разными способами. Мы начнем с обсуждения основной модели (рис. 6.1).

Основная модель имеет *конечное управление*, *ленту*, которая разделена на *ячейки*, и *головку ленты*, которая сканирует одну ячейку ленты в один прием. Лента имеет крайнюю левую ячейку, но простирается в бесконечность в правую сторону. Каждая ячейка может содержать ровно один из конечного числа *символов ленты*. Первоначально n крайних левых ячеек для некоторого конечного n содержит *входную цепочку*, строку символов ленты, называемых *входными символами*. Остальные ячейки до бесконечности содержат *пробел* — специальный символ ленты, который не является входным символом.

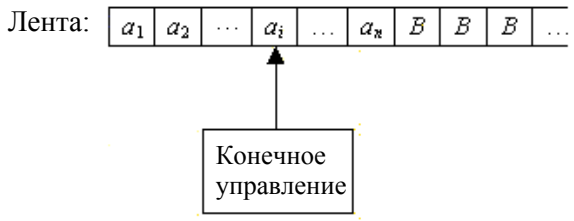

Рис. 6.1.

В один *такт*, зависящий от символа, сканируемого головкой ленты, и состояния конечного управления, машина Тьюринга

1) изменяет состояние;

2) печатает символ ленты, не являющийся пробелом, в сканируемой ячейке ленты, замещая то, что было там записано;

3) сдвигает свою головку влево или вправо на одну ячейку.

***Определение* 6.1.** *Машина Тьюринга* (Tm) является формальной системой: $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$, где Q — конечное *множество состояний*; Γ — конечное *множество допустимых символов ленты*, один из них, обычно обозначаемый буквой B, есть *пробел*; $\Sigma \subseteq \Gamma \setminus \{B\}$ — *множество входных символов*; $\delta : Q \times \Gamma \to Q \times (\Gamma \setminus \{B\}) \times \{L, R\}$ — *функция следующего такта* (*движения*), для некоторых аргументов может быть не определена[8]; $q_0 \in Q$ — *начальное состояние*; $F \subseteq Q$ — *множество конечных состояний*.

***Определение* 6.2.** *Конфигурацией* машины Тьюринга назовем тройку (q, α, i), где $q \in Q$ — текущее состояние машины Тьюринга; $\alpha \in (\Gamma \setminus \{B\})^*$ — строка, являющаяся непустой частью ленты; i — целое, определяющее позицию головки ленты, отсчитываемую от левого конца ленты.

Заметим, что если головка ленты покидает ячейку, она должна напечатать непустой символ в этой ячейке, так что лента всегда содержит непустой блок символов (α — этот блок) с бесконечным числом пробелов справа от него.

Определим теперь один такт (движение) машины Тьюринга T. Пусть $(q, A_1A_2 \ldots A_n, i)$ — некоторая ее конфигурация, где $1 \le i \le n + 1$.

Случай 1. Если $1 \le i \le n$ и $\delta(q, A_i) = (p, A, R)$, то

$$(q, A_1A_2 \ldots A_n, i) \vdash_T (p, A_1A_2 \ldots A_{i-1}AA_{i+1} \ldots A_n, i + 1),$$

т.е. T печатает символ A в i-й позиции и двигается вправо.

Случай 2. Если $2 \le i \le n$ и $\delta(q, A_i) = (p, A, L)$, то

$$(q, A_1A_2 \ldots A_n, i) \vdash_T (p, A_1A_2 \ldots A_{i-1}A\, A_{i+1} \ldots A_n, i - 1),$$

т.е. T печатает символ A в i-й позиции и двигается влево, не сходя с левого конца ленты.

---

[8] Мы не позволили Tm печатать пробел ради простоты определения конфигураций. Однако Tm могла бы иметь другой символ, который трактуется точно так же, как пробел, за исключением того, что Tm разрешается печатать этот символ псевдопробела. Разумеется, никакой дополнительной мощности не появляется за счет введения такого символа. В неформальном обсуждении мы часто допускаем печать пробела, зная, что вместо него можно использовать другой, но эквивалентный ему символ.

Случай 3. Если $i = n + 1$, то головка сканирует пробел B.

а) Если при этом $\delta(q, B) = (p, A, R)$, то

$$(q, A_1A_2 \ldots A_n, n + 1) \vdash_T (p, A_1A_2 \ldots A_nA, n + 2).$$

б) Если при этом $\delta(q, B) = (p, A, L)$, то

$$(q, A_1A_2 \ldots A_n, n + 1) \vdash_T (p, A_1A_2 \ldots A_nA, n).$$

Таким образом, мы ввели отношение непосредственного следования одной конфигурации за другой. Очевидным образом можно определить степень, транзитивное и рефлексивно-транзитивное замыкания этого отношения. Будем обозначать их традиционно через $\vdash_T^n$, $\vdash_T^+$ и $\vdash_T^*$ соответственно. Если две конфигурации связаны знаком $\vdash_T^n$, то мы будем говорить, что вторая получается из первой за n движений. Соответственно запись $\vdash_T^+$ обозначает положительное число движений, а $\vdash_T^*$ — любое число движений, включая нуль.

***Определение* 6.3.** Пусть $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$ — машина Тьюринга. *Язык*, *принимаемый* машиной T есть $L = \{w \mid w \in \Sigma^*$ и $(q_0, w, 1) \vdash_T^* (q, \alpha, i)$ для некоторых $q \in F$, $\alpha \in \Gamma^*$ и $i > 0\}$.

Мы предполагаем, что данная Tm, распознающая язык L, *останавливается*, т.е. не имеет никакого следующего движения, всякий раз, как входная цепочка принимается. Но для цепочек, которые не принимаются, Tm может не остановиться.

Пример 6.1. Построим Tm, распознающую cfl $L = \{0^n1^n \mid n \geq 1\}$. Положим $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$, где $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$; $\Sigma = \{0,1\}$; $\Gamma = \{0, 1, B, X, Y\}$; $F = \{q_5\}$. Функцию δ определим следующим образом:

1. $\delta(q_0, 0) = (q_1, X, R)$.

В состоянии q_0 символ 0 заменяется на X и машина сдвигается вправо в состояние q_1 в поисках 1.

2. а) $\delta(q_1, 0) = (q_1, 0, R)$;
б) $\delta(q_1, Y) = (q_1, Y, R)$;
в) $\delta(q_1, 1) = (q_2, Y, L)$.

Оставаясь в состоянии q_1, машина продвигается вправо сквозь все нули (п. 2а) и блок Y (п. 2б). Наткнувшись на 1, заменяет ее на Y и переходит в состояние q_2, начав движение влево (п. 2в).

3. а) $\delta(q_2, Y) = (q_2, Y, L)$;
б) $\delta(q_2, X) = (q_3, X, R)$;
в) $\delta(q_2, 0) = (q_4, 0, L)$.

Оставаясь в состоянии q_2, машина продвигается влево сквозь блок Y (п. 3а). Если машина встречает X, все еще оставаясь в состоянии q_2, то больше нет нулей, которые следовало бы заменять на X, и машина переходит в состояние q_3, начиная движение вправо, чтобы убедиться, что не осталось единиц (п. 3б). Если же 0 встретился, машина переходит в состояние q_4, чтобы продолжить движение в поисках крайнего левого 0 (п. 3в).

4. а) $\delta(q_4, 0) = (q_4, 0, L)$

б) $\delta(q_4, X) = (q_0, X, R)$.

Машина движется сквозь нули (п. 4а). Если встретился X, то машина прошла самый левый нуль. Она должна, сдвинувшись вправо, превратить этот 0 в X (п. 4б). Происходит переход в состояние q_0, и процесс, только что описанный в п. 1–4, продолжается.

5. а) $\delta(q_3, Y) = (q_3, Y, R)$

б) $\delta(q_3, B) = (q_5, Y, R)$.

Машина входит в состояние q_3, когда ни одного 0 не остается (см. п. 3б). Машина должна продвигаться вправо (п. 5а) сквозь блок Y. Если встречается пробел раньше, чем 1, то ни одной 1 не осталось (п. 5б). В этой ситуации машина переходит в конечное состояние q_5 и останавливается, сигнализируя тем самым прием входной цепочки.

6. Во всех случаях, кроме 1–5, функция δ не определена.

Рассмотрим действия машины Тьюринга на входной цепочке 000111.

Табл. 6.1

Шаг	Конфигурация	Шаг	Конфигурация	Шаг	Конфигурация
1	$0\ 0\ 1\ 1\ 1$ q_0	10	$X\ X\ 0\ Y\ 1\ 1$ q_1	19	$X\ X\ X\ Y\ Y\ Y$ q_2
2	$X\ 0\ 0\ 1\ 1\ 1$ q_1	11	$X\ X\ 0\ Y\ 1\ 1$ q_1	20	$X\ X\ X\ Y\ Y\ Y$ q_2
3	$X\ 0\ 0\ 1\ 1\ 1$ q_1	12	$X\ X\ 0\ Y\ Y\ 1$ q_2	21	$X\ X\ X\ Y\ Y\ Y$ q_2
4	$X\ 0\ 0\ 1\ 1\ 1$ q_1	13	$X\ X\ 0\ Y\ Y\ 1$ q_2	22	$X\ X\ X\ Y\ Y\ Y$ q_3
5	$X\ 0\ 0\ Y\ 1\ 1$ q_2	14	$X\ X\ 0\ Y\ Y\ 1$ q_4	23	$X\ X\ X\ Y\ Y\ Y$ q_3
6	$X\ 0\ 0\ Y\ 1\ 1$ q_4	15	$X\ X\ 0\ Y\ Y\ 1$ q_0	24	$X\ X\ X\ Y\ Y\ Y$ q_3
7	$X\ 0\ 0\ Y\ 1\ 1$ q_4	16	$X\ X\ X\ Y\ Y\ 1$ q_1	25	$X\ X\ X\ Y\ Y\ Y$ q_3
8	$X\ 0\ 0\ Y\ 1\ 1$ q_0	17	$X\ X\ X\ Y\ Y\ 1$ q_1	26	$X\ X\ X\ Y\ Y\ Y\ Y$ q_5
9	$X\ X\ 0\ Y\ 1\ 1$ q_1	18	$X\ X\ X\ Y\ Y\ 1$ q_1		

В табл. 6.1 приведены конфигурации в виде цепочек символов ленты с маркером состояния под сканируемым символом (в конфигурациях 25 и 26 маркер состояния находится под символом пробела).

§ 6.2. Методы построения машин Тьюринга

Машина Тьюринга может “программироваться” во многом так же, как программируются вычислительные машины. Роль программы играет функция δ. В этом параграфе мы представим коллекцию приемов программирования машины Тьюринга, которые помогут лучше узнать ее возможности.

6.2.1. Конечное управление в роли запоминающего устройства. Конечное управление может использоваться для запоминания конечного количества информации. Именно: состояние записывается как пара элементов, причем один осуществляет управление, а другой запоминает символ. Подчеркнем, что этот прием используется только концептуально. Никакой модификации основной модели машины Тьюринга не подразумевается.

Пример 6.2. Пусть $T = (Q, \{0, 1\}, \{0, 1, B\}, \delta, [q_0, B], F)$, где $Q = \{[q_0, 0], [q_0, 1], [q_0, B], [q_1, 0], [q_1, 1], [q_1, B]\}$, $F = \{[q_1, B]\}$, т.е. здесь Q записано как $\{q_0, q_1\} \times \{0, 1, B\}$.

Запрограммируем распознавание языка, состоящего из цепочек, в которых первый символ повторно не встречается в той же самой цепочке.

Отметим, что такой язык является регулярным.

Построим функцию δ следующим образом:

1. а) $\delta([q_0, B], 0) = ([q_1, 0], 0, R)$;
б) $\delta([q_0, B], 1) = ([q_1, 1], 1, R)$.

Машина запоминает сканируемый символ во второй компоненте обозначения состояния и сдвигается вправо. Первой компонентой становится q_1.

2. а) $\delta([q_1, 0], 1) = ([q_1, 0], 1, R)$;
б) $\delta([q_1, 1], 0) = ([q_1, 1], 0, R)$.

Если машина помнит 0 и видит 1 или, наоборот, помнит 1 и видит 0, то она продолжает движение вправо.

3. а) $\delta([q_1, 0], B) = ([q_1, B], 0, L)$;
б) $\delta([q_1, 1], B) = ([q_1, B], 0, L)$.

Машина входит в конечное состояние $[q_1, B]$, если она встречает символ пробела раньше, чем достигает второй копии самого левого символа. Если же машина достигает пробела в состоянии $[q_1, 0]$ или $[q_1, 1]$, то она принимает входную цепочку. Для состояния $[q_1, 0]$ и символа 0 или для состояния $[q_1, 1]$ и символа 1 функция δ не определена, так что, если машина когда-нибудь видит запомненный символ, она останавливается, не принимая.

В общем случае можно допустить произвольное фиксированное число компонент, причем все, кроме одной, предназначены для запоминания информации.

6.2.2. Многодорожечные ленты. Мы можем подразумевать, что лента машины Тьюринга разделена на k дорожек для любого конечного k. На рис. 6.2

представлено такое устройство для $k = 3$. Прием состоит в том, что символы на ленте рассматриваются как наборы из k элементов, по одному на каждой дорожке.

Пример 6.3. Представим, что лента, показанная на рис. 6.2, является лентой машины Тьюринга, которая принимает двоичный код целых чисел, значение которых больше 2. Этот код записан на верхней дорожке и ограничен слева и справа символами ¢ и $.

Рис. 6.2.

Задача машины — определить, является ли закодированное число простым. Мы не будем строить фактическую программу этой машины, а только обсудим способ кодирования информации на 3-дорожечной ленте и дадим словесное описание ее работы.

В начальный момент на верхней дорожке, как уже сказано, находится двоичный код тестируемого числа с упомянутыми ограничителями слева и справа. Две другие дорожки пусты. Таким образом, входными символами являются тройки $[¢, B, B]$, $[0, B, B]$, $[1, B, B]$ и $[\$, B, B]$. Они отождествляются соответственно с ¢, 0, 1 и $. Символ пробела можно представить как $[B, B, B]$.

Чтобы проверить, находится ли на ее ленте код простого числа, машина, во-первых, записывает двоичный код числа 2 на второй дорожке и копирует первую дорожку на третью. Затем вторая дорожка вычитается из третьей столько раз, сколько можно, фактически оставляя остаток на третьей дорожке. Если остаток есть нуль, то число на первой дорожке не простое. Если остаток не нуль, то число на второй дорожке увеличивается на 1. Если теперь вторая дорожка равна первой, число на первой дорожке простое, поскольку оно не делится без остатка на любое число между единицей и самим собой. Если вторая дорожка меньше первой, вся операция повторяется для нового числа на второй дорожке и восстановленной копии первой дорожки на третьей.

На рис. 6.2 машина определяет, является ли число 47 простым. В показанный момент она делит 47 на 5 и уже дважды вычитание 5 было выполнено, так что на третьей дорожке содержится промежуточный результат: 37.

6.2.3. Отметка символов является полезным приемом, чтобы увидеть, как машина Тьюринга распознает языки, предложения которых состоят из повторяющихся строк, например $\{ww \mid w \in \Sigma^*\}$, $\{wcy \mid w, y \in \Sigma^*, w \neq y\}$ или $\{ww^R \mid w \in \Sigma^*\}$.

Этот прием полезен также тогда, когда необходимо сравнивать длины подцепочек, как, например, в языках $\{a^i b^i \mid i \geq 1\}$ или $\{a^i b^j c^k \mid i \neq j \text{ или } j \neq k\}$. Прием состоит в том, что вводится дополнительная дорожка на ленте машины Тью-

ринга, которая содержит пробел или символ √. Последний используется для отметки того, что символ, расположенный под ним, уже рассмотрен машиной в одном из предыдущих сравнений.

Пример 6.4. Построим машину Тьюринга, которая распознает язык $L=\{wcw \mid w\in \{a, b\}^+\}$. Положим $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$, где $Q = \{[q, d] \mid q\in \{q_1, q_2,..., q_9\}, d\in \{a, b, B\}\}$; вторая компонента состояния используется для того, чтобы запомнить входной символ; $\Sigma = \{[B, d] \mid d\in \{a, b, c\}\}$; $\Gamma = \{[X, d] \mid X\in \{B, √\}, d\in \{a, b, c, B\}\}$, $q_0 = [q_1, B]$, $F = \{[q_9, B]\}$. Символ пробела представляется как $[B, B]$, символ a идентифицируется с $[B, a]$, символ b идентифицируется с $[B, b]$ и символ c идентифицируется с $[B, c]$.

Определим функцию δ следующим образом:

1. $\delta([q_1, B], [B, d]) = ([q_2, d], [√, d], R)$ для $d\in \{a, b\}$.

Машина отмечает сканируемый символ на ленте, запоминает его в конечном управлении и начинает движение вправо, переходя в новое состояние с первой компонентой q_2.

2. $\delta([q_2, d], [B, e]) = ([q_2, d], [B, e], R)$ для $d,e\in \{a, b\}$.

Машина продолжает двигаться вправо по непроверенным символам в поисках символа c.

3. $\delta([q_2, d], [B, c]) = ([q_3, d], [B, c], R)$ для $d\in \{a, b\}$.

По нахождению символа c машина входит в новое состояние с первой компонентой q_3.

4. $\delta([q_3, d], [√, e]) = ([q_3, d], [√, e], R)$ для $d,e\in \{a, b\}$.

Машина движется вправо по проверенным символам.

5. $\delta([q_3, d], [B, d]) = ([q_4, B], [√, d], L)$ для $d\in \{a, b\}$.

Если машина “видит” неотмеченный символ, равный запомненному в конечном управлении, то отмечает его и начинает движение влево, переходя в новое состояние с первой компонентой q_4 и “забывая” символ, который она только что отметила. В противном случае дальнейшее движение не определено: машина останавливается, не принимая.

6. $\delta([q_4, B], [√, d]) = ([q_4, B], [√, d], L)$ для $d\in \{a, b\}$.

Машина движется влево по проверенным символам.

7. $\delta([q_4, B], [B, c]) = ([q_5, B], [B, c], L)$.

Машина встречает символ c, фиксирует этот факт переходом в новое состояние $[q_5, B]$ и продолжает движение влево.

8. $\delta([q_5, B], [B, d]) = ([q_6, B], [B, d], L)$ для $d\in \{a, b\}$.

Если символ непосредственно слева от c не отмечен, то машина фиксирует этот факт переходом в новое состояние $[q_6, B]$ и, продолжая двигаться влево, начинает поиск самого левого из непомеченных символов.

9. $\delta([q_6, B], [B, d]) = ([q_6, B], [B, d], L)$ для $d\in \{a, b\}$.

Машина продолжает движение влево сквозь непомеченные символы.

10. $\delta([q_6, B], [\surd, d]) = ([q_1, B], [\surd, d], R)$ для $d \in \{a, b\}$.

Машина встречает помеченный символ и сдвигается вправо, чтобы взять следующий символ для сравнения. Первой компонентой состояния снова становится q_1. С этого момента цикл запоминания очередного непроверенного символа и его поиска на правой половине входной цепочки повторяется.

11. $\delta([q_5, B], [\surd, d]) = ([q_7, B], [\surd, d], R)$ для $d \in \{a, b\}$.

В состоянии $[q_5, B]$ машина окажется сразу после прохождения символа c при попятном движении (см. п. 7). Если символ непосредственно слева от c отмечен, то это значит, что все символы на левой половине цепочки проверены. Остается проверить, все ли символы и на правой половине тоже отмечены. Если это так, то цепочка слева от c равна цепочке справа от c, и машина останавливается, принимая входную цепочку. Иначе — машина останавливается, не принимая эту цепочку.

12. $\delta([q_7, B], [B, c]) = ([q_8, B], [B, c], R)$.

Машина сдвигается вправо от c, фиксируя переходом в новое состояние $[q_8, B]$ режим проверки отметки всех символов на правой половине цепочки.

13. $\delta([q_8, B], [\surd, d]) = ([q_8, B], [\surd, d], R)$ для $d \in \{a, b\}$.

Машина двигается вправо через отмеченные символы на правой половине цепочки.

14. $\delta([q_8, B], [B, B]) = ([q_9, B], [\surd, B], L)$.

Если машина находит пробел $[B, B]$, она останавливается и принимает входную цепочку. Если машина находит неотмеченный символ, когда первая компонента ее состояния — q_8, она останавливается, не принимая входной цепочки.

6.2.4. Сдвиг символов ленты. Машина Тьюринга может освобождать место на своей ленте, сдвигая все непустые символы на конечное число ячеек вправо. Она может затем вернуться к освобожденным ячейкам и печатать символы по своему выбору. Если пространство доступно, машина может сдвигать блоки символов влево подобным же образом.

Пример 6.5. Построим часть машины Тьюринга $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$, которая время от времени имеет необходимость сдвигать символы на две ячейки вправо. Пусть Q содержит состояния вида $[q, A_1, A_2]$ для $q \in \{q_1, q_2\}$ и $A_1, A_2 \in \Gamma$. Пусть B — пробел, а X — специальный символ, не используемый машиной нигде, кроме как в процессе сдвига. Мы предполагаем, что машина начинает процесс сдвига в состоянии $[q_1, B, B]$. Относящаяся к этому часть функции δ определяется следующим образом:

1. $\delta([q_1, B, B], A_1) = ([q_1, B, A_1], X, R)$ для $A_1 \in \Gamma \setminus \{B, X\}$.

Машина запоминает первый прочитанный символ в третьей компоненте ее состояния, печатает на его месте символ X и начинает движение вправо.

2. $\delta([q_1, B, A_1], A_2) = ([q_1, A_1, A_2], X, R)$ для $A_1, A_2 \in \Gamma \setminus \{B, X\}$.

Машина запоминает прочитанный символ в третьей компоненте ее состояния, сдвигая перед этим третью компоненту на место второй, печатает на его месте символ X и двигается вправо.

3. $\delta([q_1, A_1, A_2], A_3) = ([q_1, A_2, A_3], A_1, R)$ для $A_1, A_2, A_3 \in \Gamma \setminus \{B, X\}$.

Машина запоминает прочитанный символ в третьей компоненте ее состояния, сдвигая перед этим третью компоненту на место второй, печатает на его месте символ A_1 из прежней второй компоненты состояния и двигается вправо. Очевидно, что символ A_1 будет напечатан двумя ячейками правее той позиции, в которой он находился прежде. Заметим, что вторая и третья компоненты состояния играют роль буфера, через который проходят задерживаемые символы. Поскольку буфер рассчитан на два элемента, он обеспечивает задержку каждого символа на то время, пока машина проходит две позиции на ленте.

4. $\delta([q_1, A_1, A_2], B) = ([q_1, A_2, B], A_1, R)$ для $A_1, A_2, A_3 \in \Gamma \setminus \{B, X\}$.

Когда на ленте виден пробел, запомненный во второй компоненте символ помещается на ленту.

5. $\delta([q_1, A_1, B], B) = ([q_2, B, B], A_1, L)$ для $A_1 \in \Gamma \setminus \{B, X\}$.

Последний запомненный символ из второй компоненты состояния помещается на ленту, после чего начинается движение влево с целью выйти на крайнюю правую свободную ячейку. Она помечена, как и все другие освобожденные ячейки, символом X.

6. $\delta([q_2, B, B], A) = ([q_2, B, B], A, L)$ для $A \in \Gamma \setminus \{B, X\}$.

Машина двигается влево, пока не выходит на символ X. После этого машина перейдет в состояние, которое, как мы предполагаем, существует в множестве состояний Q, и возобновит другие свои действия.

6.2.5. Моделирование. Пусть B — автомат, который с входной цепочкой w последовательно проходит конфигурации $C_1, C_2, \ldots, C_n$. Неформально, мы говорим, что автомат A *моделирует* автомат B, если автомат A с входной цепочкой w проходит последовательно конфигурации, представляющие $C_1, C_2, \ldots, C_n$.

Возможно, что автомат A будет входить в другие конфигурации в промежутках между конфигурациями, представляющими конфигурации автомата B.

Понятие моделирования полезно при доказательстве, что автомат одного типа может распознавать язык, принимаемый автоматом некоторого другого типа. Чтобы автомат A моделировал автомат B, он должен быть способен, во-первых, по представлению конфигурации C_i вычислять представление C_{i+1}; во-вторых, автомат A должен определять, является ли конфигурация C_i автомата B принимающей. В этом случае A тоже должен принимать его собственную входную цепочку.

Попутно заметим, что для целей моделирования часто бывает удобно представлять конфигурацию машины Тьюринга в виде $\alpha qX\beta$, где α и β являются ленточными цепочками, X — символ ленты, q — состояние. В конфигурации $\alpha qX\beta$ состояние есть q, $\alpha X\beta$ — непустая часть ленты, а X — символ, сканируемый головкой ленты. Пример моделирования недетерминированной машины Тьюринга с помощью детерминированной приводится в § 6.4.

6.2.6. Диагонализация. Другое полезное понятие — *диагонализация*. Оно может быть использовано для того, чтобы показать, что существует язык, при-

нимаемый автоматом типа 2, который не принимается никаким автоматом типа 1. Диагонализация характеризуется несколькими отличительными чертами.

а) Должно существовать кодирование всех автоматов типа 1, входные символы которых выбираются из одного и того же алфавита Σ. Пример того, как это кодирование может быть сделано, приводится в следующей главе.

б) Должен строиться автомат A типа 2 с входными цепочками из Σ^*. Входная цепочка автомата A трактуется как кодирование некоторого автомата B типа 1 с его собственной входной цепочкой.

в) Автомат A должен моделировать автомат B и определять, принимает B свою собственную входную цепочку или нет. Если автомат B принимает входную цепочку, то автомат A не принимает, и наоборот.

г) Всегда истинно, что язык, принимаемый автоматом A типа 2, не принимается никаким автоматом типа 1. Действительно, предположим, что B является таким автоматом типа 1, который принимает язык, принимаемый автоматом A. Автомат B имеет кодирование $w \in \Sigma^*$. Предположим, что автомат B принимает цепочку w. Тогда автомат A не принимает цепочку w, и наоборот. В любом случае, автоматы A и B не могут принимать один и тот же язык.

Было бы уместно дать следующее разумное объяснение термина “диагонализация”. Можно перенумеровать все слова в множестве Σ^* следующим образом. Взять сначала самые короткие, и среди слов равной длины использовать некоторый лексикографический порядок. Затем мы можем занумеровать автоматы типа 1 согласно номерам, приписанным их кодам. Автомат A принимает i-е слово тогда и только тогда, когда i-е слово не принимается i-м автоматом. Вообразим бесконечную матрицу, элемент которой (i, j) есть 1, если i-й автомат принимает j-е слово, и 0 — в противном случае. Автомат A принимает i-е слово, когда элемент (i, i) есть 0. Отсюда и термин “диагонализация”.

6.2.7. Подпрограммы. Одна машина Тьюринга может быть “подпрограммой” другой машины Тьюринга при весьма общих условиях. Если T_1 должна быть подпрограммой для T_2, мы требуем, чтобы их состояния не пересекались. Но с состояниями других подпрограмм состояния T_1 могут пересекаться.

Чтобы “вызвать” T_1, машина T_2 входит в начальное состояние машины T_1. Правила машины T_1 являются частью правил машины T_2. Кроме того, из состояния остановки машины T_1, машина T_2 входит в свое собственное состояние и продолжает работу.

Пример 6.6. Опишем неформально машину Тьюринга T_3, которая вычисляет $n!$ Именно: машина, запущенная с входной цепочкой вида 01^n0, должна заканчивать работу с кодом вида $0^{n+2}1^{n!}$ на непустой части ее ленты.

Машина T_3 использует машину T_2 в качестве подпрограммы, которая выполняет умножение. Именно: машина T_2, запущенная с входной цепочкой вида

$01^i0^j1^k$ на ее ленте и ленточной головкой на крайнем левом нуле блока из j нулей, останавливается с результатом вида $01^i0^j1^{ik}$.

Из своей начальной конфигурации машина T_3, двигаясь вправо, проходит блок единиц и последний нуль, печатает 1 и движется влево. В этот момент на ленте машины T_3 находится цепочка 01^n01. Далее машина T_3 вызывает подпрограмму T_2. Когда управление вернется к машине T_3, ее лента будет содержать цепочку 01^n01^n0. Затем из своего текущего состояния машина T_3 проверяет, остается ли в первом блоке единиц, по крайней мере, три единицы. Если это так, она изменяет крайнюю правую единицу в этом блоке на нуль и возвращается в состояние, из которого она вызывает подпрограмму T_2.

После второго вызова подпрограммы T_2 на ленте машины T_3 появится цепочка $01^{n-1}001^{n(n-1)}0$. После третьего вызова на ленте будет последовательность $01^{n-2}0001^{n(n-1)(n-2)}0$, и вызовы T_2 будут повторяться до тех пор, пока после $(n-1)$-го вызова на ленте не появится цепочка $0110^{n-1}0001^{n(n-1)(n-2)\ldots 2}0$. В это время первый блок единиц имеет меньше трех 1, так что машина T_3 изменяет их на нули и останавливается при состоянии ленты $0^{n+2}1^{n(n-1)(n-2)\ldots 2}0 = 0^{n+2}1^{n!}0$.

В свою очередь, машина T_2 сама использует машину T_1 в качестве своей подпрограммы, которая добавляет первый блок единиц, не изменяя его, ко второму. При построении машины T_1 пригодится метод отметки посредством символов "отключения проверки". Машина T_2 вызывает подпрограмму T_1 один раз для каждой единицы в первом блоке, отключая ее проверку. Таким образом осуществляется умножение.

§ 6.3. Машина Тьюринга как процедура

До сих пор мы определяли машину Тьюринга как распознающее устройство. Но можно рассматривать машину Тьюринга и как процедуру. Например, если мы желаем описать процедуру для определения того, является ли число простым, мы могли бы построить машину Тьюринга, которая принимает множество всех простых чисел. Рассматриваем мы эту машину в данном случае как распознаватель или как процедуру — дело вкуса.

Если желательно рассмотреть процедуру манипуляции над строками символов, то можно свести ее к проблеме распознавания при помощи построения другой машины Тьюринга, которая принимает пары строк, разделенных специальным символом. Эта машина принимает данную пару точно тогда, когда процедура превратила бы первую строку в паре во вторую и остановилась. Мы оставим доказательство того факта, что по данной процедуре можно найти соответствующий распознаватель и наоборот, в качестве упражнения для читателя. Большинство необходимых для этого идей содержит §1.2.

Машина Тьюринга в примере 6.1 используется как распознаватель. Заметим, что независимо от того, какова входная цепочка, эта машина со временем дос-

тигнет условия, при котором для состояния конечного управления и сканируемого символа функция δ не определена. В таком случае машина Тьюринга останавливается и никакие дальнейшие ее движения невозможны. Если язык принимается машиной Тьюринга, которая останавливается на всех входных цепочках, то говорят, что язык *рекурсивен*.

Следует подчеркнуть, что существуют языки, которые принимаются машинами Тьюринга, не останавливающимися для некоторых цепочек, не содержащихся в языке, но которые не принимаются никакими машинами Тьюринга, останавливающимися на всех входных цепочках. Язык, который может быть распознан некоторой машиной Тьюринга, называется *рекурсивно перечислимым множеством* (recursively enumerable set — res). В следующей главе будет показано, что рекурсивно перечислимые множества являются в точности языками, порождаемыми грамматиками типа 0.

Когда машина Тьюринга рассматривается как процедура и оказывается, что она останавливается для всех входных цепочек, то говорят, что такая процедура есть *алгоритм*.

Есть процедуры, для которых не существует никакого соответствующего алгоритма. Примером их является процедура для определения, порождает ли контекстно-зависимая грамматика (csg), по крайней мере, одну терминальную цепочку. Можно построить машину Тьюринга, которая по заданной csg будет порождать все возможные терминальные цепочки в некотором лексикографическом порядке. К каждой цепочке эта машина Тьюринга применяет алгоритм, данный в гл. 2, чтобы увидеть, порождается ли данная цепочка грамматикой. Если эта грамматика порождает хотя бы одно слово, машина найдет его и остановится в конечном состоянии. Но если язык, порождаемый этой грамматикой, пуст, то машина будет продолжать порождать слова и проверять их вечно. Имеет место факт, что не существует машины Тьюринга, которая останавливается на каждой входной цепочке и определяет для каждой csg, является ли язык, порождаемый этой грамматикой, пустым. Другими словами, проблема распознавания непустоты контекстно-зависимого языка алгоритмически неразрешима.

В дополнение к рассмотрению машины Тьюринга как распознающего устройства или процедуры мы можем рассматривать ее как *средство определения функций*. Пусть $f(n)$ — функция, отображающая положительные целые в положительные целые и пусть $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$ — машина Тьюринга. Если для каждого целого n имеет место $(q_0, 1^n, 1) \overset{*}{\underset{T}{\vdash}} (p, 1^{f(n)}, 1)$ для $p \in F$, то говорят, что машина вычисляет функцию $f(n)$.

Если некоторая машина Тьюринга T вычисляет функцию $f(n)$ для каждого n, то говорят, что функция $f(n)$ *рекурсивна*.

Если $f(n)$ определена для не всех n, то говорят, что $f(n)$ — *частичная функция.*

Если некоторая машина Тьюринга T вычисляет функцию $f(n)$ всякий раз, как $f(n)$ определена, но может не остановиться для тех n, для которых $f(n)$ не определена, то $f(n)$ — *частично рекурсивная функция*.

§ 6.4. Модификации машин Тьюринга

Одна из причин, по которой машины Тьюринга принимаются в качестве общей модели вычисления, состоит в том, что модель, с которой мы имеем дело, инвариантна по отношению ко многим модификациям, которые, казалось бы, увеличивают вычислительную мощность устройства. В этом параграфе даются доказательства некоторых из этих теорем об эквивалентности.

6.4.1. Машина Тьюринга с бесконечной лентой в обе стороны обозначается как и первоначальная модель: $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$. Как подразумевает ее название — лента бесконечна не только вправо, но и влево. Конфигурация такого устройства, как и прежде, есть (q, α, i), где $q \in Q$ — состояние, $\alpha \in \Gamma^+$ — непустая строка символов, но i (и это не обычно) — *положение головки ленты относительно левого конца строки* α. Другими словами, $i = 1$, если машина сканирует самый левый символ цепочки α; $i = 2$, если она сканирует второй символ, и т.д. Предполагается, что имеется бесконечно много пустых ячеек не только справа, но и слева от цепочки α. Таким образом, возможно, что $i = 0$, и в этом случае машина сканирует пробел непосредственно слева от цепочки α.

Отношение $\vdash_T$, связывающее две конфигурации, из которых вторая получается из первой при помощи единственного движения, определяется как для первоначальной модели со следующими исключениями для $i \le 1$:

a) если $\delta(q, X) = (p, Y, L)$, то $(q, X\alpha, 1) \vdash_T (p, Y\alpha, 0)$;

b) если $\delta(q, B) = (p, Y, R)$, то $(q, \alpha, 0) \vdash_T (p, Y\alpha, 2)$;

c) если $\delta(q, B) = (p, Y, L)$, то $(q, \alpha, 0) \vdash_T (p, Y\alpha, 0)$.

Здесь B — пробел, а $Y \ne B$. Начальной конфигурацией является $(q_0, w, 1)$. В отличие от первоначальной модели нет никакого левого конца ленты, который "выпадает", так что она может двигаться влево сколь угодно далеко.

Степень, транзитивное замыкание и рефлексивно-транзитивное замыкание отношения $\vdash_T$ определяются аналогично.

Теорема 6.1. *Если язык L распознается машиной Тьюринга* (*Tm*) *с бесконечной в обе стороны лентой, то он распознается Tm с полубесконечной лентой.*

Доказательство. Пусть $T_2 = (Q_2, \Sigma_2, \Gamma_2, \delta_2, q_2, F_2)$ — Tm с лентой, бесконечной в обе стороны. Мы построим T_1 — машину Тьюринга, моделирующую машину T_2 и имеющую ленту, которая бесконечна только вправо. Эта лента имеет две дорожки. На одной представляются ячейки ленты машины T_2, расположенные вправо от начального положения головки, включая и саму ячейку, сканируемую вначале. На другой — ячейки, расположенные слева от начальной в порядке их удаления от начальной позиции. Если мы припишем начальной ячейке машины T_2 номер 0, ячейкам, которые справа, — номера 1, 2,..., а тем ячейкам, которые слева, припишем номера –1, –2, ... , то связь между лентами машин T_2

(рис. 6.3, *а*) и T_1 (рис. 6.3, *б*) может быть представлена так, как показано на этом рисунке.

а

A_0	A_1	A_2	A_3	A_4	…

б

A_0	A_1	A_2	A_3	A_4	…
¢	A_{-1}	A_{-2}	A_{-3}	A_{-4}	…

Рис. 6.3.

Первая ячейка на ленте машины T_1 будет содержать на нижней дорожке символ ¢, указывающий, что это — самая левая ячейка. Конечное управление машины T_1 будет содержать информацию относительно того, сканирует ли машина T_2 символ, находящийся на верхней или на нижней дорожке.

Очевидно, что машина T_1, моделирующая машину T_2, может быть построена. Когда машина T_2 находится справа от начальной позиции, машина T_1 работает на верхней дорожке. Когда машина T_2 находится слева от начальной позиции, машина T_1 работает на нижней дорожке, двигаясь в направлении, противоположном направлению движения машины T_2. Входные символы машины T_1 — символы с пробелом на нижней дорожке и входным символом машины T_2 на верхней дорожке. Такой символ идентифицируется с соответствующим входным символом машины T_2.

Теперь выполним формальное построение $T_1 = (Q_1, \Sigma_1, \Gamma_1, \delta_1, q_1, F_1)$.

Положим $Q_1 = \{q_1\} \cup \{[q, D] \mid q \in Q_2, D \in \{U, L\}\}$. Здесь вторая компонента состояния указывает, работает ли машина T_1 на верхней или нижней дорожке[9].

Ленточные символы $\Gamma_1 = \{[X, Y] \mid X \in \Gamma_2, Y \in \Gamma_2 \cup \{¢\}, ¢ \notin \Gamma_2\}$. Если B — пробел на ленте машины T_2, то $[B, B]$ — пробел на ленте машины T_1.

Входные символы $\Sigma_1 = \{[a, B] \mid a \in \Sigma_2\}$. Мы идентифицируем a с $[a, B]$.

Конечные состояния $F_1 = \{[q, D] \mid q \in F_2, D \in \{U, L\}\}$.

Функцию δ_1 определим следующим образом:

1. Для любого $a \in \Sigma_2$ полагаем

$\delta_1(q_1, [a, B]) = ([q, U], [X, ¢], R)$, если $\delta_2(q_2, a) = (q, X, R)$.

Если первое движение машины T_2 происходит вправо, то машина T_1 печатает ¢ на нижней дорожке, чтобы отметить левый край ленты, устанавливает вторую компоненту состояния на U и двигается вправо. Первая компонента состояния машины T_1 фиксирует состояние машины T_2. При этом машина T_1 печатает символ X на верхней дорожке, если машина T_2 печатает его на своей ленте.

2. Для любого $a \in \Sigma_2$ полагаем

$\delta_1(q_1, [a, B]) = ([q, L], [X, ¢], R)$, если $\delta_2(q_2, a) = (q, X, L)$.

[9] Здесь символ L обозначает lower (нижняя) и его не следует путать с L в составе значений δ, где L обозначает left (влево).

Если первое движение машины T_2 происходит влево, то машина T_1 печатает $\not\subset$ на нижней дорожке, чтобы отметить левый край ленты, устанавливает вторую компоненту состояния на L и двигается вправо. Первая компонента состояния T_1 фиксирует состояние T_2. При этом машина T_1 печатает символ X на верхней дорожке, если машина T_2 печатает его на своей ленте.

3. Для любого $[X, Y] \in \Gamma_1$ с $Y \neq \not\subset$ и $D \in \{L, R\}$ полагаем

$$\delta_1([q, U], [X, Y]) = ([p, U], [Z, Y], D), \text{ если } \delta_2(q, X) = (p, Z, D).$$

4. Для любого $[X, Y] \in \Gamma_1$ с $Y \neq \not\subset$ и $D \in \{L, R\}$ полагаем

$$\delta_1([q, L], [X, Y]) = ([p, L], [X, Z], D), \text{ если } \delta_2(q, Y) = (p, Z, \overline{D}).$$

Если $D = L$, то $\overline{D} = R$. Если $D = R$, то $\overline{D} = L$.

Машина T_1 моделирует машину T_2 на своей нижней дорожке. Направление движения головки машины T_1 противоположно направлению движения машины T_2.

5. Полагаем, что

$$\delta_1([q, U], [X, \not\subset]) = \delta_1([q, L], [X, \not\subset]) = ([p, E], [Y, \not\subset], R), \text{ если } \delta_2(q, X) = (p, Y, D);$$

$E = U$, если $D = R$; $E = L$, если $D = L$.

Машина T_1 моделирует движение машины T_2 на ячейке, первоначально сканированной машиной T_2. Затем машина T_1 работает на верхней или нижней дорожке в зависимости от направления, в котором движется машина T_2. В этой позиции машина T_1 будет всегда двигаться вправо.

Доказательство того, что обе машины принимают один и тот же язык, предоставляется читателю в качестве упражнения.

6.4.2. Многоленточная машина Тьюринга состоит из конечного управления с k ленточными головками, по одной на каждой ленте (рис. 6.4).

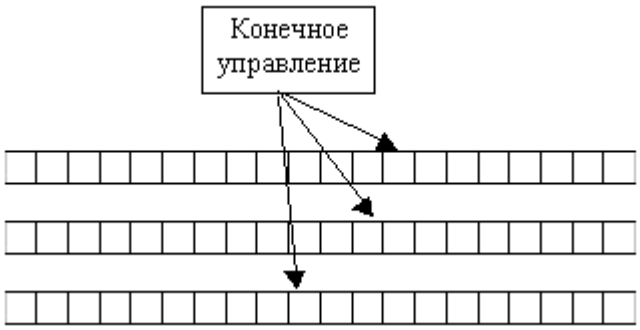

Рис. 6.4.

Каждая лента бесконечна в обоих направлениях. При одном движении, зависящем от состояния конечного управления и сканируемого символа каждой из ленточных головок, машина может:

1) изменить состояние;

2) напечатать новый символ на каждой из сканируемых ячеек;

3) передвинуть каждую из ее ленточных головок независимо друг от друга на одну ячейку влево, вправо или оставить ее на том же месте.

Сначала входная цепочка имеется только на первой ленте, а все другие ленты пусты. Мы не будем определять это устройство более формально, предоставляя это читателю.

Теорема 6.2. *Если язык L принимается многоленточной машиной Тьюринга, то он принимается одноленточной машиной Тьюринга.*

Доказательство. Пусть язык L принимается машиной Тьюринга T_1 с k лентами. Построим одноленточную машину Тьюринга T_2 с $2k$ дорожками, по две для каждой из лент машины T_1. На одной дорожке записывается содержимое соответствующей ленты машины T_1, а другая — пустая, за исключением маркера в ячейке, содержащей символ и сканируемой соответствующей головкой машины T_1. Такое устройство для моделирования трех лент посредством одной иллюстрируется рис. 6.5.

Конечное управление машины T_2 запоминает, какие маркеры головок машины T_1 находятся слева, а какие — справа от головки T_2. Состояния машины T_1 тоже запоминаются в конечном управлении машины T_2.

Головка 1		×					
Лента 1	A_1	A_2	…		…	A_m	
Головка 2				×			
Лента 2	B_1	B_2	…		…	B_m	
Головка 3	×						
Лента 3	C_1	C_2	…		…	C_m	

Рис. 6.5.

Чтобы моделировать движение машины T_1, машина T_2 должна посетить каждую ячейку с маркером головки, регистрируя по очереди символ, сканируемый соответствующей головкой T_1. Когда машина T_2 проходит через маркер головки, она должна уточнять направление, в котором следует искать этот маркер. После сбора всей необходимой информации машина T_2 определяет движение машины T_1. Затем машина T_2 посещает по очереди каждый из маркеров головок снова, изменяя маркированные ячейки и сдвигая маркеры на одну ячейку, если необходимо. Конечно, если новое состояние является принимающим, то машина T_2 принимает входную цепочку.

Пример 6.7. Посмотрим, насколько легче многоленточной машине Тьюринга распознать язык $L = \{ww^R \mid w \in \{0,1\}^*\}$, чем одноленточной.

Чтобы распознать язык L на одноленточной машине Тьюринга, ее головка ленты должна двигаться вперед и назад по ленте, отмечая и сравнивая символы с обоих концов. Это подобно процессу из примера 6.4.

Чтобы распознать язык L с помощью двухленточной машины Тьюринга, входная цепочка копируется на вторую ленту. Затем цепочка на одной ленте

сравнивается с ее копией на другой ленте в обратном порядке, и ее длина проверяется на четность.

Заметим, что для распознавания языка L однолентоточная машина делает $O(n^2)$ движений, где n — длина входной цепочки, тогда как двухленточной машине достаточно совершить только $O(n)$ движений.

6.4.3. Недетерминированная машина Тьюринга есть устройство с конечным управлением и одной бесконечной в обе стороны лентой. Для данного состояния и ленточного символа, сканируемого головкой ленты, машина имеет несколько вариантов для следующего движения. Каждый вариант состоит из нового состояния, ленточного символа, который печатается, и направления движения головки. Недетерминированная машина Тьюринга принимает входную цепочку, если какая-нибудь последовательность вариантов движений приводит к принимающему состоянию.

Теорема 6.3. *Если язык L принимается недетерминированной машиной Тьюринга T_1, то он принимается некоторой детерминированной машиной Тьюринга T_2.*

Доказательство. Для любого состояния и ленточного символа машины T_1 имеется конечное число вариантов для выбора следующего движения. Варианты могут быть занумерованы числами 1, 2, ... Пусть r — максимальное число вариантов для любой пары состояние — ленточный символ. Тогда любая последовательность вариантов движений конечной длины может быть представлена последовательностью цифр от 1 до r. Не все такие последовательности могут представлять варианты движений, поскольку в некоторых конфигурациях вариантов может быть меньше, чем r.

Можно построить детерминированную машину Тьюринга T_2, моделирующую машину T_1. Снабдим ее тремя лентами. Первая будет содержать входную цепочку. На второй машина T_2 будет систематически генерировать последовательность цифр от 1 до r. Конкретно: последовательности будут генерироваться, начиная с самой короткой. Среди последовательностей одинаковой длины они генерируются в числовом порядке.

Для каждой последовательности, сгенерированной на ленте 2, машина T_2 копирует вход на ленту 3 и затем моделирует машину T_1 на ленте 3, используя последовательность ленты 2 для того, чтобы диктовать движения машине T_1.

Если машина T_1 входит в принимающее состояние, то машина T_2 также принимает. Если имеется последовательность вариантов, ведущая к приему, то она в конце концов будет сгенерирована на ленте 2. Будучи смоделирована, машина T_2 будет принимать входную цепочку. Но если никакая последовательность вариантов движений машины T_1 не ведет к приему входной цепочки, то машина T_2 не примет ее.

Заметим, что это доказательство можно обобщить, чтобы показать, как моделировать недетерминированную многоленточную машину Тьюринга обычной моделью машины Тьюринга.

6.4.4. Двумерная машина Тьюринга является еще одной модификацией машины Тьюринга, которая не увеличивает ее мощности. Это устройство состоит из обычного конечного управления, но лента разбита на бесконечное число ячеек, расположенных в двух измерениях. В зависимости от состояния и сканируемого символа устройство изменяет состояние, печатает новый символ и передвигает ленточную головку в одном из четырех направлений. Первоначально входная цепочка находится на одной строке, а головка находится на левом конце вводной цепочки.

В любое время только конечное число строк имеет какие-нибудь непустые символы на них, и каждая из этих строк имеет только конечное число непустых символов.

Рассмотрим например, конфигурацию ленты, показанную на рис. 6.6, *а*. Мы можем очертить прямоугольник вокруг непустых символов, как показано на этом рисунке, и этот прямоугольник можно записать строка за строкой на одномерной ленте, как показано на рис. 6.6, *б*.

а

B	B	B	a_1	B	B	B
B	B	a_2	a_3	a_4	a_5	B
a_6	a_7	a_8	a_9	B	a_{10}	B
B	a_{11}	a_{12}	a_{13}	B	a_{14}	a_{15}
B	B	a_{16}	a_{17}	B	B	B

б

$*BBBa_1BBB*BBa_2a_3a_4a_5B*a_6a_7a_8a_9Ba_{10}B*Ba_{11}a_{12}a_{13}Ba_{14}a_{15}*BBa_{16}a_{17}BBB*$

Рис. 6.6.

Символы * разделяют строки. Один из символов помечается как сканируемый головкой. Для пометки можно использовать дополнительную дорожку ленты. Если при данном движении головка остается в пределах представленного прямоугольника, то подогнать положение головки нетрудно. Если головка выходит за границу прямоугольника при движении в вертикальном направлении, добавляют еще одну строку пробелов к левому или правому концу линейного представления. Если головка покидает прямоугольник через правую или левую границу, длина каждой представленной строки должна быть увеличена на единицу. При этом может пригодиться метод “сдвига”.

Описанный подход легко обобщить на n-мерные ленты.

6.4.5. Машина Тьюринга с входной лентой только для чтения и одной или несколькими лентами памяти для записи/чтения также заслуживает упоминания. Ее движение зависит от сканируемого входного символа, но она не может печатать на входной ленте. Обычно считается, что входная лента имеет концевые маркеры, так что головка входной ленты может всегда оставаться на входной цепочке, границы которой она не может сама отмечать.

Если входная головка может двигаться в двух направлениях, то устройство называется *машиной Тьюринга типа off-line*. Если входная головка никогда не движется влево, то это *машина Тьюринга типа on-line*. Ясно, что машины этих двух типов являются вариантами многоленточных машин Тьюринга. Они могут моделировать любую многоленточную машину Тьюринга.

§ 6.5. Ограниченные машины Тьюринга, эквивалентные основной модели

До сих пор рассматривались обобщения основной модели машины Тьюринга. Как мы видели, эти обобщения не увеличивают вычислительную мощность этой модели.

Данную главу заключим обсуждением некоторых моделей, которые с первого взгляда могут показаться менее мощными, чем обычные машины Тьюринга, но на самом деле имеют точно такую же мощность. По большей части эти модели будут вариациями базисного магазинного автомата, рассмотренного в гл. 5. Попутно отметим, что магазинный автомат можно рассматривать как недетерминированную машину Тьюринга с двумя лентами: одной только для чтения, причем ее входная головка не может двигаться влево, и другой — лентой памяти с весьма специфическим ограничением для ее ленточной головки. Всякий раз, как головка ленты памяти движется влево, она должна печатать пробел. Таким образом, лента памяти справа от головки — всегда пустая. Строго говоря, машине Тьюринга запрещено печатать настоящий пробел. Взамен можно было бы ввести другой специальный символ с теми же самыми правилами, какие имеются для пробела. Предоставим читателю убедиться в том, что такая модель эквивалентна магазинному автомату, введенному в гл. 5.

6.5.1. Детерминированная машина с двумя магазинными лентами — это детерминированная машина Тьюринга с входной цепочкой только для чтения и двумя магазинными лентами памяти. Если головка магазинной ленты движется влево, то она печатает “пробел”.

Лемма 6.1. *Произвольная одноленточная машина Тьюринга может быть смоделирована детерминированной машиной с двумя магазинными лентами*.

Доказательство. Достаточно убедиться в том, что символы слева от головки машины Тьюринга, которая моделируется, могут запоминаться на одной магазинной ленте, а те, что справа, — на другой.

6.5.2. Машина со счетчиками есть машина Тьюринга, алфавиты лент памяти которой содержат только два символа, например Z и B. Кроме того, символ Z появляется в ячейке, первоначально сканируемой ленточной головкой, и никогда не может появиться ни в какой другой. Число i можно запомнить путем передвижения головки ленты на i ячеек вправо от Z. Предполагается, что машины этого типа могут печатать пробел. Запомненное число может быть увеличено или уменьшено передвижением головки ленты вправо или влево.

Пример машины со счетчиком показан на рис. 6.7. Символы ¢ и $ обычно используются в качестве концевых маркеров на входной ленте. Здесь Z — непустой символ.

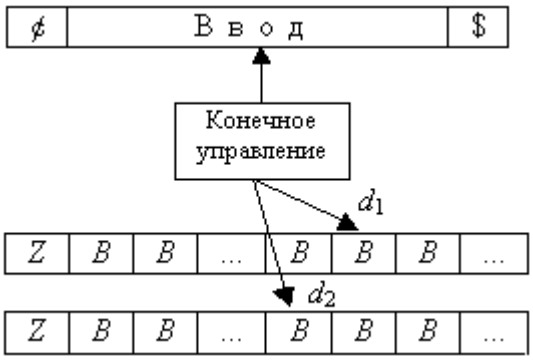

Рис. 6.7.

Конфигурация машины со счетчиками может быть описана при помощи состояния, положения входной головки и расстояний головок лент памяти от символа Z (на рис. 6.7 они обозначены символами d_1 и d_2). Назовем эти расстояния *отсчетами* на лентах. Машина со счетчиками может фактически только запоминать отсчет на каждой ленте и указывать, является ли этот отсчет нулевым.

Лемма 6.2. *Машина с четырьмя счетчиками может моделировать произвольную машину Тьюринга*.

Доказательство. Согласно лемме 6.1 достаточно показать, что две ленты со счетчиками могут моделировать одну магазинную ленту.

Пусть магазинная лента имеет $k-1$ непустых ленточных символов Z_1, Z_2, ..., Z_{k-1}. Тогда мы можем однозначно представить магазинную ленту $Z_{i_1}Z_{i_2}...Z_{i_m}$ при помощи "счетчика" $j = i_m + k\, i_{m-1} + k^2\, i_{m-2} + ... + k^{m-1} i_1$. Предположим, что j запоминается на одном счетчике, т.е. головка ленты находится на j ячеек правее непустого символа. Пусть головка второго счетчика также находится на непустом символе.

Предположим, что символ Z_r печатается на вершине (правом конце) магазинной ленты $Z_{i_1}Z_{i_2}...Z_{i_m}$. Счетчик, связанный с $Z_{i_1}Z_{i_2}...Z_{i_m}Z_r$, равен $kj + r$. Чтобы установить этот новый счетчик, машина со счетчиком повторно сдвигает головку первого счетчика на одну ячейку влево, а головку второго — на k ячеек вправо. Когда головка первого счетчика достигнет непустого символа, второй счетчик будет представлять kj. Добавить r к этому счетчику просто. Наоборот, чтобы удалить верхний символ Z_{i_m} магазина, необходимо заменить значение счетчика j на целую часть j / k. Для этого надо повторно продвигать влево головку первого счетчика на k ячеек, и после каждого такого продвижения головки второго счетчика продвигать влево только на одну ячейку. Если в очередной раз окажется невозможно продвинуть головку первого счетчика ровно на k ячеек влево из-за того, что непустая ячейка будет достигнута раньше, чем закончится продвижение на k ячеек, то в этот момент на втором счетчике образуется требуемое значение.

Чтобы закончить описание моделирования, остается рассказать, как узнать, какой символ находится на вершине магазинной ленты, моделируемой двумя счетчиками. Если установлено значение j на одном счетчике, то машина может скопировать j в другой счетчик, вычисляя j по модулю k в своем конечном управлении. Заметим, что j по модулю k это и есть i_m.

Теорема 6.4. *Машина с двумя счетчиками может моделировать произвольную машину Тьюринга*.

Доказательство. Согласно лемме 6.2 достаточно показать, как моделировать четыре счетчика двумя. Пусть четыре счетчика имеют отсчеты i, j, k и m. Все четыре значения можно установить на одном счетчике в виде числа $n = 2^i3^j5^k7^m$. Поскольку 2, 3, 5 и 7 — простые числа, то значения i, j, k и m могут быть однозначно восстановлены из n.

Чтобы увеличить i, j, k или m на единицу, достаточно умножить n на 2, 3, 5 или 7 соответственно.

Если мы имеем второй счетчик, установленный на нуль, то можем сдвигать головку этого счетчика на 2, 3, 5 или 7 ячеек вправо каждый раз, как головка первого счетчика продвигается на одну ячейку влево. Когда первый счетчик достигнет нулевого отсчета, второй будет содержать отсчеты $2n$, $3n$, $5n$ или $7n$ соответственно.

Чтобы уменьшить i, j, k или m на единицу, достаточно при помощи подобного процесса разделить n на 2, 3, 5 или 7 соответственно.

Мы должны также показать, как машина с двумя счетчиками может определить следующее движение машины с четырьмя счетчиками. Входная головка машины с двумя счетчиками будет всегда в той же самой точке на ее входной ленте, в какой была бы входная головка машины с четырьмя счетчиками. Состояние машины с четырьмя счетчиками может быть запомнено в конечном управлении двухсчетчиковой машины. Следовательно, чтобы определить движение четырехсчетчиковой машины, двухсчетчиковая должна только определить, какие отсчеты из i, j, k, m равны нулю. Передавая n из одного счетчика в другой, конечное управление двухсчетчиковой машины может определить, делится ли n на 2, 3, 5, 7 или какое-нибудь их произведение.

Теорема 6.5. *Каждая машина Тьюринга может быть смоделирована машиной Тьюринга с входной лентой только для чтения и лентой памяти с двумя символами* (*пробел и другой символ*) *при условии*, *что машина может печатать пробелы*.

Доказательство. Большая часть доказательства будет оставлена читателю. Прием состоит в том, чтобы закодировать каждый из k символов памяти при помощи r бинарных символов, где $2^r \geq k$. Головка ленты машины Тьюринга может посетить каждый из r бинарных символов, представляющих первоначальный символ, чтобы определить, каким он был.

Замечательно, что может быть доказана теорема более сильная, чем теорема 6.5.

Теорема 6.6. *Каждая машина Тьюринга может моделироваться машиной Тьюринга с входной лентой только для чтения и лентой памяти с двумя символами*: 0 (*пробел*) *и* 1. *Машина Тьюринга может печатать* 0 *или* 1, *где был найден 0, но не может печатать 0 там, где находилась* 1.

Доказательство мы оставляем читателю. Прием состоит в том, чтобы смоделировать последовательные конфигурации исходной машины Тьюринга на ленте новой машины. Символы ленты, конечно, кодируются в бинарных символах. Каждая конфигурация копируется и попутно производятся необходимые изменения, отражающие движение исходной машины.

Помимо бинарного кодирования первоначального символа машина Тьюринга, моделирующая исходную, нуждается в ячейках, чтобы указывать (а) позицию головки в конфигурации, которая копируется, и (б) что бинарное представление символа уже скопировано.