Глава 7

МАШИНЫ ТЬЮРИНГА: ПРОБЛЕМА ОСТАНОВКИ, ЯЗЫКИ ТИПА 0

§ 7.1. Универсальная машина Тьюринга

В этой главе мы покажем, что существует машина Тьюринга U, которая по заданному коду произвольной машины Тьюринга T и кодированию входной цепочки x будет моделировать поведение машины T с входной цепочкой x. Такая машина U называется *универсальной машиной Тьюринга*. Ее можно рассматривать как вычислительную машину общего назначения, которая достаточно мощна для того, чтобы моделировать любую вычислительную машину, включая саму себя.

Мы покажем также, что не существует алгоритма (т.е. машины Тьюринга, которая останавливается на всех входных цепочках), который мог бы определить для произвольной машины Тьюринга T и произвольной ее входной цепочки x, остановится ли когда-нибудь машина T с входной цепочкой x. Этот отрицательный результат интенсивно используется как аргумент для того, чтобы показать, что многие проблемы, относящиеся к различным классам языков, являются рекурсивно (т.е. алгоритмически) неразрешимыми.

Будет также показано, что имеются рекурсивно перечислимые множества, которые не являются рекурсивными. Другими словами, есть множества, которые распознаются машинами Тьюринга, но не такими, которые останавливаются для всех входных цепочек.

Наконец, будет доказана основная теорема об эквивалентности языков типа 0 и множеств, распознаваемых машинами Тьюринга.

Покажем, что универсальная машина Тьюринга существует путем действительного ее построения.

Прежде всего мы должны условиться относительно кодирования машин Тьюринга и кодирования их входных цепочек. Поскольку машина Тьюринга T_1 может иметь любое число допустимых символов ленты, мы предполагаем, что все они будут кодироваться при помощи символов 0 и 1. Очевидно, что для каждой Tm T_1 существует Tm T_2 с ленточными символами 0 и 1 и одним дополнительным символом ленты B (пробел), которая принимает точно те строки из множества $\{0, 1\}^*$, которые являются кодами слов, принимаемых машиной T_1. Принимая это во внимание, достаточно спроектировать универсальную машину Тьюринга для машин Тьюринга с одинаковыми ленточными алфавитами $\{0, 1, B\}$.

Машина Тьюринга с тремя допустимыми ленточными символами может быть полностью определена при помощи лл. 7.1.

Табл. 7.1

Состояние	Входной символ		
	B	0	1
1	—	—	2, 0, *R*
2	3, 1, *L*	3, 1, *L*	2, 1, *R*
3	4, 0, *R*	4, 0, *R*	3, 1, *L*
4	—	—	—

Поскольку машина Тьюринга может иметь произвольно большое число состояний и поскольку мы имеем только фиксированное число допустимых символов ленты, то кодируем состояния в виде 1, 11, 111, и т.д.

Один из способов закодировать таблицу состояний состоит в том, чтобы разметить некоторое число блоков, равное числу состояний, а затем разбить каждый блок на три подблока. Состоянию *i* будет соответствовать *i*-й блок, а три подблока будут относиться к входным символам *B*, 0 и 1 соответственно. Блоки будут отделяться двумя символами *c*, а подблоки одним символом *c*. Начало и конец таблицы будет отмечаться тремя символами *c*.

Если в машине Тьюринга *T*, которая кодируется, $\delta(i, a) = (j, b, D)$, то подблок, соответствующий состоянию i и входному символу a, будет содержать j единиц, за которыми следует символ $D \in \{L, R\}$, а за ним $b \in \{0, 1\}$. Если $\delta(i, a)$ не определено, то соответствующий подблок будет содержать единственный нуль. Таким образом, кодировка табл. 7.1 оказалась бы такой, как приводимая ниже запись:

ccc 0c0c11R0cc
111L1c111L1c11Rcc
1111R0c1111R0c111L1cc
0c0c0ccc

Код 11*R*0 в подблоке, соответствующем состоянию 1 и входному символу 1, означает то, что машина *T*, будучи в состоянии 1 и сканируя символ 1, будет заменять 1 на 0, двигаться вправо и входить в состояние 2.

Любое состояние, в котором для всех трех допустимых символов ленты переходные состояния не определены, интерпретируется как принимающее состояние. Таково состояние 4 в табл. 7.1. Предполагается, что после приема входной цепочки машина не делает больше никаких движений.

В любом не принимающем состоянии, по крайней мере, для одного допустимого символа ленты следующее состояние должно быть определено.

В роли начального состояния всегда используется состояние 1.

Хотя мы использовали только пять символов, чтобы закодировать машину Тьюринга, показанную в табл. 7.1, наша универсальная машина Тьюринга будет использовать 12 символов ленты. Дополнительные символы появляются из того соображения, что она будет иметь двухдорожечную ленту. Нижняя дорожка будет использовать символы *c*, 0, 1, *L*, *R* и *B*, в то время как верхняя — символы *m* и *B*.

Табл.7.2

Сост.	*B*						*m*						Пояснения
	0	1	c	*L*	*R*	*B*	0	1	c	*L*	*R*	*B*	
A	Двигаться вправо					—	—	—	*B*, *R*	—	—	—	Найти маркер в области данных
B	Двигаться вправо					—	C_0, *L*	C_0, *L*	—	—	—	C_B, *L*	
C_B	Двигаться влево					—	—	—	$D_B, \binom{B}{c}, R$	—	—	—	Найти маркер в области таблицы состояний
C_0	Двигаться влево					—	—	—	$D_0, \binom{B}{c}, R$	—	—	—	
C_1	Двигаться влево					—	—	—	$D_1, \binom{B}{c}, R$	—	—	—	
D_B	*V*, *L*	$E, \binom{m}{1}, L$	—	—	—	—							Найти подблок, соотвествущий входному символу
D_0	*R*	*R*	D_B, *R*	*R*	*R*	—							
D_1	*R*	*R*	D_0, *R*	*R*	*R*	—							
E	*L*	*L*	*F*, *L*	*L*	*L*	—							Найти состояние 1 и отметить. В текстовом описании этот маркер — m_2
F	*E*, *L*	*E*, *L*	*G*, *L*	*E*, *L*	*E*, *L*	—							
G	*E*, *L*	*E*, *L*	*H*, *R*	*E*, *L*	*E*, *L*	—							
H	—	—	*I*, *R*	—	—	—							
I	—	—	$J, \binom{m}{c}, R$	—	—	—							
J	Двигаться вправо					—	—	$K_L, \binom{B}{1}, R$	—	—	—	—	Начало подпрограммы усновки следующего состояния

Сост.	*B*						*m*						Пояснения
	0	1	*c*	*L*	*R*	*B*	0	1	*c*	*L*	*R*	*B*	
K_L	—	$M_L, \binom{m}{1}, L$	—	T_L, R	T_R, R	—							
M_L	Двигаться влево					—	—	—	$N_L, \binom{B}{c}, R$	—	—	—	
N_L	R	R	M_L, R	R	R	—	—	N_R, R	—	—	—	—	m_2 *слева от* m_1
P_L	N_L, R	N_L, R	$S_L, \binom{m}{c}, R$	N_L, R	N_L, R	—	—	N_R, R	—	—	—	—	
S_L	Двигаться вправо					—	—	$K_L, \binom{B}{1}, R$	—	—	—	—	
K_R	—	$M_R, \binom{m}{1}, R$	—	T_L, R	T_R, R	—							
M_R	Двигаться вправо					—	—	—	$N_R, \binom{B}{c}, R$	—	—	—	
N_R	R	R	P_R, R	R	R	—							m_2 справа от m_1
P_R	N_R, R	N_R, R	$S_R, \binom{m}{c}, L$	N_R, R	N_R, R	—							
S_R	Двигаться влево					—	—	$K_N, \binom{B}{1}, R$	—	—	—	—	
T_L	T_{L-0}, R	T_{L-1}, R	—	—	—	—							Запоминание символа для печати
T_R	T_{L-0}, R	T_{L-1}, R	—	—	—	—							
T_{L-0}	Двигаться вправо					—	$U, \binom{B}{0}, L$	$U, \binom{B}{0}, L$	—	—	—	$U, \binom{B}{0}, L$	Найти маркер в области данных
T_{L-1}	Двигаться вправо					—	$U, \binom{B}{1}, L$	$U, \binom{B}{1}, L$	—	—	—	$U, \binom{B}{1}, L$	
T_{R-0}	Двигаться вправо					—	$U, \binom{B}{0}, R$	$U, \binom{B}{0}, R$	—	—	—	$U, \binom{B}{0}, R$	
T_{R-1}	Двигаться вправо					—	$U, \binom{B}{1}, R$	$U, \binom{B}{1}, R$	—	—	—	$U, \binom{B}{1}, R$	

Окончание табл. 7.2

Сост.	*B*						*m*						Пояснения
	0	1	*c*	*L*	*R*	*B*	0	1	*c*	*L*	*R*	*B*	
U	$C_0, \binom{m}{0}, L$	$C_1, \binom{m}{1}, L$	—	—	—	$C_B, \binom{m}{B}, L$							Установить маркер
V	*L*	*L*	*W*, *L*	*L*	*L*	—							Проверить, является ли состояние, в котором случилась остановка, принимающим
W	*V*, *L*	*V*, *L*	X_1, *R*	*V*, *L*	*V*, *L*	—							
X_1	—	—	X_2, *R*	—	—	—							
X_2	X_3, *R*	—	—	—	—	—							
X_3	—	—	X_4, *R*	—	—	—							
X_4	X_5, *R*	—	—	—	—	—							
X_5	—	—	X_6, *R*	—	—	—							
X_6	*Y*, *R*	—	—	—	—	—							
Y	—	—	—	—	—	—							Принять

Примечание. Элементы таблицы имеют следующие значения. Тройка из состояния, ленточного символа и *L* или *R* указывает следующее состояние, символ ленты, который печатать, и направление движения. Пара из состояния и *L* или *R* указывает следующее состояние и направление движения, при этом ленточный символ остается неизменным. Символы *L* или *R* и слова “Двигаться влево” или “Двигаться вправо” указывают, что машина Тьюринга движется влево или вправо, не изменяя состояния и символа ленты. Прочерк обозначает ситуацию, которая никогда не случается. Состояние *Y* является принимающим. Следовательно, никакое движение из состояния *Y* невозможно.

Теперь неформально опишем универсальную машину Тьюринга (U). Вход в нее, как сказано, устроен на двухдорожечной ленте. Нижняя дорожка будет содержать кодировку некоторой машины Тьюринга (T), за которой будет следовать цепочка из нулей и единиц, представляющая ее входные данные. Данные отделяются от кодирования машины тремя последовательными c. Первоначально верхняя дорожка будет вся заполнена символами B, за исключением двух ячеек: третьего символа c в цепочке ccc в начале кода машины и первой ячейки данных — см. ниже:

m m

ccc блок сост. 1 cc блок сост. 2 cc ... cc блок последнего состояния $\underbrace{ccc\ 0110...}_{\text{данные}}$

Универсальная машина Тьюринга U будет моделировать движения, которые бы предпринимала Tm T на входной цепочке, представленной как данные на ленте машины U, следующим образом.

Во-первых, машина U передвигает свою головку вправо, пока она не обнаружит маркер в области данных над входным символом, сканируемым машиной T. Этот символ, назовем его A, запоминается в конечном управлении машины U, которая с этого момента начинает движение влево, пока не достигнет маркера, регистрирующего текущее состояние машины T (отмечающего соответствующий блок в коде таблицы машины T). Машина U удаляет этот маркер (заменяет его символом B), передвигается вправо к подблоку, соответствующему символу A, и помещает маркер над первым символом в этом подблоке при условии, что он есть 1. Если же он — 0, то машина U останавливается, поскольку нет никакого следующего движения у машины T. В последующем этот маркер подблока будем называть m_1.

Предположим, что первый символ был 1. Тогда машина U движется влево, пока не находит цепочку ccc. Затем машина U движется вправо, помечая крайнее правое из этих трех c. Этот маркер назовем m_2. Машина U продолжает продвигаться вправо, пока не будет найден маркер m_1. После этого машина U входит в подпрограмму, которая попеременно передвигает маркер m_1 на одну 1 вправо, а маркер m_2 — на один блок вправо. Чтобы отличить маркеры, которые оба есть m, машина U будет запоминать в своем конечном управлении, какой маркер она видела последним. Когда машина U сдвигает m_1 на символ, который не является 1, маркер m_2 располагается над символом c, который как раз перед блоком, соответствующим следующему состоянию машины T. В этой точке машина U удаляет маркер m_1 и записывает в своем конечном управлении символ, который машина T будет печатать, и направление, в котором машина T будет двигать свою головку ленты. Затем машина U снова движется вправо в область данных и находит маркер, который указывает место головки ленты машины T. Символ, находящийся под маркером, заменяется на символ, запомненный в конечном управлении, а маркер сдвигается на одну ячейку в направлении, также зафиксированном в конечном управлении. Таким образом машина U смоделировала одно движение машины T.

Далее машина U запоминает новый символ, сканируемый машиной T, в своем конечном управлении, начинает двигаться влево, пока не достигнет маркера m_2, регистрирующего состояние машины T, и повторяет процесс, который был только что описан.

Если машина T останавливается с этими конкретными данными, то машина U, в точности воспроизведя все движения машины T, тоже останавливается, а в области данных будет зафиксировано финальное содержание ленты машины T.

Когда машина T останавливается, машина U может сообщить, находится ли машина T в принимающем состоянии или нет.

Если машина T не останавливается, то машина U тоже не останавливается, т.е. не принимает.

Так наша универсальная машина Тьюринга моделирует машину Тьюринга T. Детальное устройство универсальной машины Тьюринга, которую мы описали неформально, дано в табл. 7.2.

Отметим, что эта универсальная машина Тьюринга имеет 12 ленточных символов, но может моделировать только машину Тьюринга с двумя допустимыми символами ленты. Но можно построить эквивалентную универсальную машину Тьюринга, которая будет использовать только два допустимых символа ленты. Для этого каждый допустимый символ ленты надо закодировать блоком из четырех символов 0 или 1. Часть первоначальной ленты с данными будет использовать четыре ячейки вместо одной непустой ячейки на оригинальной входной ленте машины T.

§ 7.2. Неразрешимость проблемы остановки

Проблема остановки машины Тьюринга формулируется следующим образом: дана машина Тьюринга в произвольной конфигурации со строкой непустых ленточных символов конечной длины. Остановиться ли она в конце концов?

Говорят, что эта проблема рекурсивно не разрешима в том смысле, что не существует алгоритма, который для каждой Tm и каждой конфигурации определял бы, остановится ли машина когда-нибудь. Это совсем не значит, что мы не можем определить, остановится ли конкретная Tm в конкретной ситуации.

При описании универсальной машины Тьюринга мы имели кодирование для любой Tm с ленточными символами 0, 1 и B. Кодированием была цепочка из $\{0, 1, c, L, R\}^*$. Мы можем перенумеровать все такие цепочки, перечисляя их в порядке возрастания длины. Цепочки одинаковой длины упорядочиваются в соответствии со значением строки по основанию 5. Предполагается, что эти 5 символов играют роль целых 0, 1, 2, 3, 4 в каком-нибудь порядке. Аналогичным образом цепочки из множества $\{0, 1\}^*$ могут быть тоже упорядочены. Первыми цепочками являются ε, 0, 1, 00, 01, 10, 11, 000, 001, ... Таким образом, имеет смысл говорить об i-й цепочке в множестве $\{0, 1\}^*$.

Если мы предположим, что каждая цепочка из множества $\{0, 1, c, L, R\}^*$ является машиной Тьюринга (некоторые цепочки будут образованы неправильно — они рассматриваются как Tm без каких-нибудь движений), то также имеет смысл говорить о j-й машине Тьюринга, т.е. о машине, представленной j-й цепочкой из множества $\{0, 1, c, L, R\}^*$.

Рассмотрим язык $L_1 = \{x_i \mid x_i$ не принимается Tm $T_i\}$. Ясно, что язык L_1 не мог бы приниматься никакой Tm. Если бы это не было так, то существовала бы некоторая машина Тьюринга, скажем T_j, которая бы принимала язык L_1. Возьмем, цепочку x_j. Если $x_j \in L_1$, то по определению языка x_j не принимается машиной T_j. С другой стороны, машина T_j распознает язык L_1, стало быть принимает $x_j \in L_1$. Наше допущение привело к противоречию. Если $x_j \notin L_1$, то x_j не принимается машиной T_j, но тогда по определению языка x_j принимается машиной T_j. Опять противоречие. Остается признать, что язык L_1 не принимается никакой Tm.

Предположим, что мы имели бы алгоритм (т.е. машину Тьюринга, которая всегда останавливается) для определения, остановится ли когда-нибудь машина Тьюринга в данной конфигурации. Обозначим этот алгоритм T_0. Тогда мы могли бы построить машину Тьюринга T, которая принимает язык L_1, а это противоречило бы только что установленному факту. Машина T действовала бы следующим образом:

1. Пусть $x \in L_1$ на ее входе. Прежде всего она перечисляет предложения x_1, x_2, ... до тех пор, пока не обнаружит, что некоторое $x_i = x$. Таким образом Tm T определяет, что x является i-м предложением в этом перечислении.

2. Затем Tm T генерирует код машины Тьюринга T_i.

3. Управление теперь передается машине T_0, которая может определить, останавливается Tm T_i с входной цепочкой x_i или нет.

4. Если установлено, что Tm T_i не останавливается с входной цепочкой x_i (т.е. Tm T_i не принимает x_i), то машина T останавливается и принимает.

5. Если же установлено, что Tm T_i останавливается с входной цепочкой x_i, то управление передается универсальной машине Тьюринга, которая моделирует машину T_i с входной цепочкой x_i.

6. Поскольку, как было выяснено на предыдущем шаге, Tm T_i останавливается с входной цепочкой x_i, то универсальная машина Тьюринга остановится и определит, принимает машина T_i цепочку x_i или нет. В любом случае Tm T останавливается, принимая x_i в случае, когда Tm T_i цепочку x_i не принимает, и наоборот, отвергая ее, если Tm T_i ее принимает.

Итак, наше предположение о том, что существует машина Тьюринга, которая всегда останавливается и решает проблему остановки произвольной машины Тьюринга, привела нас к противоречию, состоящему в том, что мы сумели построить машину Тьюринга, распознающую язык L_1. Это дает возможность сформулировать следующее утверждение.

Теорема 7.1. *Не существует алгоритма (машины Тьюринга, которая гарантированно останавливается) для определения, остановится ли в конце концов произвольная машина Тьюринга, начиная в произвольно заданной конфигурации.*

Доказательство вытекает из подходящей формализации вышеприведенного рассуждения.

Для многих проблем не существует разрешающего алгоритма, в частности и для решения некоторых проблем, касающихся теории языков.

§ 7.3. Класс рекурсивных множеств

Мы можем теперь показать, что класс рекурсивных множеств является собственным подмножеством рекурсивно перечислимых множеств. Другими словами, существует множество, предложения которого могут быть распознаны машиной Тьюринга, которая не останавливается на некоторых предложениях не из этого множества, но не могут быть распознаны никакой машиной Тьюринга, которая всегда останавливается.

Примером такого множества является дополнение множества L_1, о котором шла речь в предыдущем параграфе. Прежде, чем доказать это, дадим две леммы.

Лемма 7.1. *Если множество рекурсивно, то его дополнение рекурсивно.*

Доказательство. Если $L \subseteq \Sigma^*$ — рекурсивное множество, то существует машина Тьюринга T, гарантированно останавливающаяся, которая принимает язык L. Можно предполагать, что после принятия входной цепочки Tm T не делает больше никаких движений. Построим другую машину Тьюринга — T_1, у которой одно принимающее состояние: q. Правила Tm T_1 включают все правила машины T, так что Tm T_1 моделирует Tm T. Кроме того, функция δ Tm T_1 доопределяется для непринимающих состояний и допустимых символов ленты, для которых дальнейшее движение не определено, движением, переводящим Tm T_1 в принимающее состояние q. В состоянии q машина T_1 останавливается, принимая входную цепочку, которая первоначальной машиной T не принимается.

Так Tm T_1 моделирует Tm T до тех пор, пока Tm T не останавливается. Если машина T останавливается в непринимающем состоянии, она, конечно, не принимает свою входную цепочку, но Tm T_1 делает еще одно движение в состояние q и принимает. Ясно, что Tm T_1 принимает язык $\Sigma^* \setminus L$. Что и требовалось доказать.

Лемма 7.2. *Пусть $x_1, x_2, \ldots$ — эффективное перечисление всех предложений над некоторым конечным алфавитом Σ, а $T_1, T_2, \ldots$ — эффективное перечисление всех машин Тьюринга с символами ленты, выбранными из некоторого конечного алфавита, включающего Σ. Пусть $L_2 = \{ x_i \mid x_i$ принимается машиной $T_i \}$.*

Утверждается, что L_2 — рекурсивно перечислимое множество, дополнение которого не является рекурсивно перечислимым.

Доказательство. Предложения языка L_2 могут приниматься машиной Тьюринга T, которая не обязательно останавливается на предложениях, не принадлежащих языку L_2, и действует следующим образом.

Для данного предложения x машина T перечисляет цепочки x_1, x_2, ... до тех пор, пока она не находит цепочку $x_i = x$, тем самым определяя, что x является i-й цепочкой в перечислении.

Затем Tm T генерирует Tm T_i и передает управление универсальной машине Тьюринга, которая моделирует Tm T_i с входной цепочкой x_i.

Если Tm T_i с входой цепочкой x_i останавливается и принимает, то Tm T тоже останавливается, принимая. Если Tm T_i останавливается и отвергает x_i, то Tm T тоже останавливается, отвергая. Наконец, если Tm T_i не останавливается, то Tm T тоже не останавливается.

Таким образом, множество L_2 — рекурсивно перечислимо, поскольку оно принимается Tm T.

Кроме того, множество $\overline{L_2}$ не может быть рекурсивно перечислимым, так как если T_j — машина Тьюринга, принимающая множество $\overline{L_2}$, то предложение x_j принадлежит множеству $\overline{L_2}$ тогда и только тогда, когда предложение x_j не принимается Tm T_j. Это противоречит утверждению, что $\overline{L_2}$ — язык, принимаемый Tm T_j. Что и требовалось доказать.

Теорема 7.2. Существует рекурсивно перечислимое множество, которое не является рекурсивным.

Доказательство. Согласно лемме 7.2 L_2 — рекурсивно перечислимое множество, дополнение которого не является рекурсивно перечислимым. Теперь, если бы L_2 было рекурсивным, то по лемме 7.1 его дополнение, $\overline{L_2}$, тоже было бы рекурсивным и, следовательно, рекурсивно перечислимым, что противоречило бы утверждению леммы 7.2. Что и требовалось доказать.

§ 7.4. Машины Тьюринга и грамматики типа 0

В этом параграфе мы докажем, что язык распознается машиной Тьюринга тогда и только тогда, когда он порождается грамматикой типа 0.

Чтобы доказать *достаточность*, мы построим недетерминированную машину Тьюринга, которая недетерминированно выбирает вывод в грамматике и смотрит, совпадает ли результат этого вывода с входной цепочкой. Если да, то машина принимает ее.

Чтобы доказать *необходимость*, мы строим грамматику, которая порождает представление терминальной строки, а затем моделирует машину Тьюринга на этой строке. Если строка принимается машиной, то строка преобразуется к терминальным символам, которые она представляет.

Теорема 7.3. *Если язык L порождается грамматикой типа* 0, *то язык L распознается машиной Тьюринга*.

Доказательство. Пусть $G = (V_N, V_T, P, S)$ — грамматика типа 0 и $L = L(G)$. Опишем неформально машину Тьюринга T, принимающую язык L. Машина T будет недетерминированной.

Пусть $T = (Q, V_T, \Gamma, \delta, q_0, F)$, где $\Gamma = V_N \cup V_T \cup \{B, \#, X\}$, причем $B, \#, X \notin V_N \cup V_T$. Мы не перечисляем всех состояний во множестве Q, но назначаем некоторые из них, как только в них возникает потребность. Мы разрешаем Tm T печатать пробел B, если необходимо.

Сначала Tm T имеет ввод $w \in V_T^*$ на ее ленте. Затем, сдвигая цепочку w на одну ячейку вправо, вставляет на освободившееся перед ней место символ #. Следом за w печатается цепочка $\#S\#$. Содержание ленты в этот момент имеет вид $\#w\#S\#$. С этого момента Tm T будет недетерминированно моделировать вывод в грамматике G, начиная с символа S. Каждая сентенциальная форма в выводе будет появляться по очереди между двумя последними ограничителями #. Если некоторый выбор движений приводит к цепочке терминалов, то она сравнивается с w. Если эти две цепочки равны, то Tm T принимает.

Формально пусть в какой-то момент Tm T имеет на своей ленте цепочку вида $\#w\#A_1A_2...A_k\#$. Машина T передвигает свою головку по цепочке $A_1A_2...A_k$, недетерминированно выбирая позицию i и константу r между 1 и максимальной длиной левой части любого правила из множества P.

Затем Tm T исследует подцепочку $A_iA_{i+1}...A_{i+r-1}$. Если она является левой частью некоторого правила из множества P, то ее можно заменить правой частью этого же правила. Машина T может быть вынуждена сдвигать $A_{i+r}A_{i+r+1}...A_k\#$ влево или вправо, чтобы освободить место или заполнить пространство, если длина правой части не равна r. При сдвиге вправо символ X используется для временного заполнения освободившегося пространства.

Из этого простого моделирования выводов в грамматике G должно быть ясно, что Tm T будет печатать на своей ленте строку вида $\#w\#\alpha\#$, где $\alpha \in V^*$, точно тогда, когда $S \overset{*}{\underset{G}{\Rightarrow}} \alpha$. Кроме того, если $\alpha = w$, то Tm T принимает. Заметим, что для реализации проверки этого равенства опять пригодится символ X. Что и требовалось доказать.

Теорема 7.4. *Если язык L распознается машиной Тьюринга*, *то язык L порождается грамматикой типа* 0.

Доказательство. Пусть язык L принимается машиной Тьюринга $T = (Q, \Sigma, \Gamma, \delta, q_0, F)$. Мы построим грамматику G, которая недетерминированно порождает две копии представления некоторого слова из множества Σ^*, а затем моделирует действие Tm T на одной из этих копий. Если Tm T принимает слово, то грамматика G превращает вторую копию в терминальную строку. Если Tm T не принимает слово, вывод никогда не дает в результате терминальную строку.

Снова мы предполагаем без потери общности рассуждений, что для каждого $q \in F$ и $a \in \Sigma$ значение $\delta(q, a)$ не определено.

Формально пусть $G = (V_N, \Sigma, P, A_1)$, где $V_N = \{[X, Y] \mid X \in \Sigma \cup \{\varepsilon\}, Y \in \Gamma\} \cup Q \cup \{A_1, A_2, A_3\}$, а

$P = \{(1)\ A_1 \to q_0 A_2,$

(2) $A_2 \to [a, a]A_2$ для каждого $a \in \Sigma$,

(3) $A_2 \to A_3$,

(4) $A_3 \to [\varepsilon, B]A_3$,

(5) $A_3 \to \varepsilon$,

(6) $q[a, C] \to [a, D]p$ для каждых $a \in \Sigma \cup \{\varepsilon\}$, $q \in Q$, $C \in \Gamma$, таких, что $\delta(q, C) = (p, D, R)$,

(7) $[b, E]q[a, C] \to p[b, E][a, D]$ для всех $C, D, E \in \Gamma$; $a, b \in \Sigma \cup \{\varepsilon\}$, $q \in Q$, таких, что $\delta(q, C) = (p, D, L)$,

(8) $[a, C]q \to qaq$, $q[a, C] \to qaq$, $q \to \varepsilon$ для каждого $a \in \Sigma \cup \{\varepsilon\}$, $C \in \Gamma$ и $q \in F\}$.

Используя правила 1 и 2, получаем вывод вида

$A_1 \overset{*}{\underset{G}{\Rightarrow}} q_0[a_1, a_1]\ [a_2, a_2]\ \ldots\ [a_k, a_k]A_2$, где $a_i \in \Sigma$, $i = 1, 2, \ldots, k$.

Предположим, что Tm T принимает цепочку $a_1 a_2 \ldots a_k$, используя не более, чем m ячеек справа от своего ввода. Тогда, используя правило 3, затем m раз правило 4 и, наконец, правило 5, продолжим предыдущий вывод. Получим

$$A_1 \overset{*}{\underset{G}{\Rightarrow}} q_0[a_1, a_1]\ [a_2, a_2]\ \ldots\ [a_k, a_k]\ [\varepsilon, B]^m.$$

Заметим, что с этого момента и впредь только правила 6 и 7 могут использоваться до тех пор, пока не порождается принимающее состояние. При этом первые компоненты в обозначениях нетерминалов никогда не изменяются, а вторые моделируют записи, производимые Tm T на ее ленте.

Индукцией по числу l движений машины T можно показать, что если $(q_0, a_1 a_2 \ldots a_k, 1) \overset{*}{\underset{T}{\vdash}} (q, X_1 X_2 \ldots X_s, r)$, то

$q_0[a_1, a_1][a_2, a_2]\ldots[a_k, a_k][\varepsilon, B]^m \overset{*}{\underset{G}{\Rightarrow}} [a_1, X_1][a_2, X_2]\ldots[a_{r-1}, X_{r-1}]q[a_r, X_r]\ldots[a_{k+m}, X_{k+m}]$,

где $a_1, a_2, \ldots, a_k \in \Sigma$; $a_{k+1} = a_{k+2} = \ldots = a_{k+m} = \varepsilon$; $X_1, X_2, \ldots, X_{k+m} \in \Gamma$; $X_{s+1} = X_{s+2} = \ldots = X_{k+m} = B$.

База. Пусть $l = 0$. Утверждение выполняется очевидным образом.

Индукционная гипотеза. Предположим, что утверждение выполняется для всех $l \le n$ $(n \ge 0)$.

Индукционный переход. Пусть Tm T выполняет следующие $n+1$ движений:

$$(q_0, a_1 a_2 \ldots a_k, 1) \overset{n}{\underset{T}{\vdash}} (q_1, X_1 X_2 \ldots X_r, j_1) \underset{T}{\vdash} (q_2, Y_1 Y_2 \ldots Y_s, j_2).$$

Тогда по индукционной гипотезе существует вывод вида

$$q_0[a_1, a_1][a_2, a_2]\ldots[a_k, a_k][\varepsilon, B]^m \overset{*}{\underset{G}{\Rightarrow}} [a_1, X_1][a_2, X_2]\ldots q_1[a_{j_1}, X_{j_1}]\ldots[a_{k+m}, X_{k+m}].$$

Судя по последнему движению, должно быть $\delta(q_1, X_{j_1}) = (q_2, Y_{j_1}, D)$ и $D = L$, если $j_2 = j_1 - 1$ или $D = R$, если $j_2 = j_1 + 1$. Соответственно при $D = R$ существует правило грамматики вида 6: $q_1[a_{j_1}, X_{j_1}] \to [a_{j_1}, Y_{j_1}]q_2$; при $D = L$ существует правило грамматики вида 7: $[a_{j_1-1}, X_{j_1-1}]q_1[a_{j_1}, X_{j_1}] \to q_2[a_{j_1-1}, X_{j_1-1}][a_{j_1}, Y_{j_1}]$.

Таким образом, еще один шаг вывода дает

$$[a_1, X_1][a_2, X_2]...q_1[a_{j_1}, X_{j_1}]...[a_{k+m}, X_{k+m}] \underset{G}{\Rightarrow} [a_1, Y_1][a_2, Y_2]...q_2[a_{j_2}, Y_{j_2}]...[a_{k+m}, Y_{k+m}],$$

где $X_i = Y_i$ для всех $i \neq j_1$. Итак, вспомогательное утверждение доказано.

Далее, если $q \in F$, то по правилам грамматики вида 8 можно получить вывод

$$[a_1, X_1][a_2, X_2]... q [a_j, X_j]...[a_{k+m}, X_{k+m}] \overset{*}{\underset{G}{\Rightarrow}} a_1a_2...a_k.$$

Итак, доказано, что если $a_1a_2...a_k$ принимается Tm T, то $a_1a_2...a_k \in L(G)$.

Чтобы завершить доказательство теоремы, остается показать, что если $A_1 \overset{*}{\underset{G}{\Rightarrow}} w$, то цепочка w принимается Tm T. Прежде всего отметим, что любой вывод и, в частности, вывод $A_1 \overset{*}{\underset{G}{\Rightarrow}} w$ может начинаться только по правилам 1–5, дающим результат вида

$$A_1 \overset{*}{\underset{G}{\Rightarrow}} q_0[a_1, a_1][a_2, a_2]...[a_k, a_k][\varepsilon, B]^m.$$

Далее могут применяться правила 6 и 7, дающие в конце концов результат вида

$$[a_1, X_1][a_2, X_2]...q[a_j, X_j]...[a_{k+m}, X_{k+m}],$$

где $q \in F$, после чего правила 8 дадут $w = a_1a_2...a_k$.

Собственно, надо показать, что движения Tm T моделируются на участке вывода

$$q_0[a_1, a_1][a_2, a_2]...[a_k, a_k][\varepsilon, B]^m \overset{*}{\underset{G}{\Rightarrow}} [a_1, Y_1][a_2, Y_2]...q[a_j, Y_j]...[a_{k+m}, Y_{k+m}],$$

где $q \in F$. Другими словами, если такой вывод имеет место, то существуют движения Tm T вида $(q_0, a_1a_2...a_k, 1) \overset{*}{\underset{T}{\vdash}} (q, Y_1Y_2...Y_{k+m}, j)$. Докажем это утверждение индукцией по l — длине вывода.

База. Пусть $l = 0$. Утверждение выполняется очевидным образом.

Индукционная гипотеза. Предположим, что утверждение выполняется для всех $l \leq n$ $(n \geq 0)$.

Индукционный переход. Пусть имеется вывод длиной $l = n + 1$. В общем случае имеем

$$q_0[a_1, a_1][a_2, a_2]...[a_k, a_k][\varepsilon, B]^m \overset{n}{\underset{G}{\Rightarrow}}$$
$$\overset{n}{\underset{G}{\Rightarrow}} [a_1, X_1][a_2, X_2]...q_1[a_{j_1}, X_{j_1}]...[a_{k+m}, X_{k+m}] \underset{G}{\Rightarrow}$$
$$\underset{G}{\Rightarrow} [a_1, Y_1][a_2, Y_2]...q[a_j, Y_j]...[a_{k+m}, Y_{k+m}],$$

где $q \in F$. Согласно индукционной гипотезе существует переход

$$(q_0, a_1a_2...a_k, 1) \overset{*}{\underset{T}{\vdash}} (q_1, X_1X_2...X_{k+m}, j_1).$$

Ясно, что последний шаг вывода мог быть выполнен только посредством правила вида 6 или 7. Если применялось правило вида 6, то $j = j_1 + 1$; если

использовалось правило вида 7, то $j = j_1 - 1$. Эти правила существуют благодаря тому, что $\delta(q_1, X_{j_1}) = (q, Y_{j_1}, D)$, где $D = R$, если $j = j_1 + 1$, или $D = L$, если $j = j_1 - 1$. При этом $X_i = Y_i$ для всех $i \neq j_1$. Благодаря этим значениям функции δ машина T совершает еще одно движение, переводящее ее в принимающую конфигурацию:

$$(q_0, a_1a_2...a_k, 1) \overset{*}{\underset{T}{\vdash}} (q_1, X_1X_2...X_{k+m}, j_1) \underset{T}{\vdash} (q, Y_1Y_2...Y_{k+m}, j),$$

где $q \in F$. Другими словами, показано, что $w = a_1a_2...a_k$ принимается Tm T.

Итак, если $w \in L(G)$, то цепочка w принимается машиной T. Теорема доказана полностью.