Глава 8

ЛИНЕЙНО ОГРАНИЧЕННЫЕ АВТОМАТЫ И КОНТЕКСТНО-ЗАВИСИМЫЕ ЯЗЫКИ

§ 8.1. Линейно ограниченные автоматы

Линейно ограниченный автомат (lba) есть недетерминированная одноленточная машина Тьюринга, которая никогда не покидает те ячейки, на которых размещен ее ввод. Формально он определяется следующим образом.

***Определение* 8.1.** *Линейно ограниченным автоматом* называется формальная система $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$, в которой Q — *множество состояний*; $q_0 \in Q$ — *начальное состояние*; $F \subseteq Q$ — *множество конечных состояний*; Γ — *алфавит допустимых символов ленты*; $\Sigma \subseteq \Gamma$ — *алфавит входных символов*, который содержит два особых символа: ¢ и \$ — *левый* и *правый маркеры*, находящиеся с самого начала на концах входной цепочки для того, чтобы предотвращать выход головки ленты за пределы участка, на котором размещается входная цепочка (считается, что маркеры могут использоваться только в этой роли: на место маркера нельзя записать какой-нибудь другой символ ленты, и никакой символ ленты не может быть заменен каким-нибудь маркером); δ — отображение типа $Q \times \Gamma \to 2^{Q \times \Gamma \times \{L, R\}}$.

Движения линейно ограниченного автомата описываются в терминах *конфигураций*. Конфигурация lba M обозначается как $(q, A_1A_2\ldots A_n, i)$, где $q \in Q$; $A_1, A_2, \ldots, A_n \in \Gamma$; i — целое, причем $1 \le i \le n$. Согласно определению $A_1 = ¢$, $A_n = \$$.

Если $(p, A, L) \in \delta(q, A_i)$ и $i > 1$, то

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

Если $(p, A, R) \in \delta(q, A_i)$ и $i < n$, то

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

Другими словами, lba M печатает символ A на месте A_i, изменяет свое состояние на p и продвигает свою головку влево или вправо, не выходя из области, в которой символы находились изначально.

Отношения на множестве конфигураций $\vdash_M$, $\vdash_M^n$, $\vdash_M^+$ или $\vdash_M^*$ определяются обычным для недетерминированных машин Тьюринга образом.

***Определение* 8.2.** *Языком, принимаемым линейно ограниченным автоматом* M, называется множество

$$\{w \mid w \in (\Sigma \setminus \{¢, \$\})^*, (q_0, ¢w\$, 1) \vdash_M^* (q, ¢\alpha\$, i), q \in F, \alpha \in \Gamma^*, 1 \le i \le n, n = |w| + 2\}.$$

О*пределение* 8.3. *Линейно ограниченный автомат* M является *детерминированным*, если $\#\delta(q, A) \le 1$ для любых $q \in Q$, $A \in \Gamma$.

Неизвестно, является ли класс множеств, распознаваемых детерминированными линейно ограниченными автоматами, строгим подклассом множеств, распознаваемых недетерминированными lba, или они совпадают. Конечно, справедливо, что любое множество, принимаемое недетерминированным линейно ограниченным автоматом, принимается некоторой детерминированной машиной Тьюринга. Однако длина ленты, требуемая этой Tm, может быть экспоненциальной, а не линейной, функцией от длины входной цепочки.

§ 8.2. Связь линейно ограниченных автоматов с контекстно-зависимыми языками

Наш интерес к недетерминированным линейно ограниченным автоматам проистекает из того факта, что класс множеств, принимаемых ими, является в точности классом контекстно-зависимых языков.

Теорема 8.1. *Если L — контекстно-зависимый язык, то язык L принимается некоторым линейно ограниченным автоматом.*

Доказательство. Пусть $G = (V_N, V_T, P, S)$ — контекстно-зависимая грамматика. Мы построим lba M, такой, что язык, принимаемый lba M, есть $L(G)$. Мы не будем вдаваться в детальное построение автомата M, поскольку оно очень сложно, а просто опишем, как он работает.

Входная лента будет иметь две дорожки. Дорожка 1 будет содержать входную строку x ($x \neq \varepsilon$) с концевыми маркерами. Дорожка 2 будет использоваться для работы.

На первом шаге lba M помещает символ S в крайнюю левую ячейку дорожки 2. Затем автомат входит в порождающую подпрограмму, которая выполняет следующие шаги:

1. Подпрограмма выбирает последовательные подстроки символов α на дорожке 2, такие, что $\alpha \to \beta \in P$.

2. Подстроки α заменяются на β, сдвигая вправо, если необходимо, символы, расположенные справа от α. Если эта операция заставляет символ быть вытолкнутым за правый маркер, автомат останавливается. Как известно, промежуточные сентенциальные формы в контекстно-зависимой грамматике не длиннее, чем выводимая терминальная цепочка. Так что, если на очередном шаге получена сентенциальная форма длиннее x, то продолжать процесс не имеет смысла, потому что все последующие сентенциальные формы будут разве лишь длиннее.

3. Подпрограмма недетерминированно выбирает, возвращаться ли к шагу 1, либо идти на выход.

4. При выходе из подпрограммы дорожка 1 все еще будет содержать строку x, в то время как дорожка 2 будет содержать некоторую строку γ, такую, что $S \overset{*}{\underset{G}{\Rightarrow}} \gamma$. Автомат M сравнивает посимвольно цепочки x и γ. Если окажется, что $x \neq \gamma$, то автомат останавливается, не принимая; если же окажется, что $x = \gamma$, то он останавливается, принимая входную цепочку.

Ясно, что если $x \in L(G)$, то найдется такая последовательность движений lba M, которая сгенерирует цепочку x на дорожке 2, и тогда автомат остановится, принимая. Аналогично, если lba M принимает цепочку x, то должна существовать последовательность движений, генерирующих цепочку x на дорожке 2. Только при таком условии lba M принимает цепочку x. Но, по построению, процесс генерации x воспроизводит вывод *этой цепочки* из S. Следовательно, $S \overset{*}{\underset{G}{\Rightarrow}} x$. Теорема доказана.

Теорема 8.2. *Если язык L принимается линейно ограниченным автоматом, то L — контекстно-зависимый язык.*

Доказательство. Построение контекстно-зависимой грамматики, которая моделирует линейно ограниченный автомат, подобно построению, описанному в теореме 7.4, в которой грамматика типа 0 строилась, чтобы моделировать движения машины Тьюринга. Нетерминалы контекстно-зависимой грамматики должны указывать не только первоначальное содержание некоторой ячейки ленты lba, но также и то, является ли эта ячейка смежной с концевым маркером

слева или справа. Такие ячейки в обозначении нетерминалов мы будем снабжать маркерами $\not\in$ и \$, обозначающими, что ячейка граничит соответственно с левым, правым или обоими концевыми маркерами. В обозначении нетерминала состояние lba должно также комбинироваться с символом, находящимся под головкой ленты. Контекстно-зависимая грамматика не может иметь отдельных символов для концевых маркеров и состояния линейно ограниченного автомата, потому что эти символы должны были бы заменяться на пустые цепочки, когда строка превращается в терминальную, а ε-порождения в csg запрещены.

Формально пусть $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ — недетерминированный lba, где $\not\in, \$ \not\in \Sigma$ и L — язык, принимаемый lba M, причем $\varepsilon \not\in L$.

Положим $G = (V_N, V_T, P, A_1)$, где

$V_N = \{A_1, A_2\} \cup \{[q, \not\in, X, a, \$], [\not\in, q, X, a, \$], [\not\in, X, a, q, \$], [q, X, a], [q, X, a, \$],$
$[X, a, q, \$], [\not\in, X, a], [X, a], [X, a, \$] \mid a \in \Sigma \setminus \{\not\in, \$\}, X \in \Gamma \setminus \{\not\in, \$\}, q \in Q\}$;

$V_T = \Sigma$; а множество P включает следующие правила:

(1) $A_1 \to [q_0, \not\in, a, a, \$]$ — моделируют начальные конфигурации вида $(q_0, \not\in a\$, 1)$;

(2.1) $[q, \not\in, X, a, \$] \to [\not\in, p, X, a, \$]$, если $(p, \not\in, R) \in \delta(q, \not\in)$;
(2.2) $[\not\in, q, X, a, \$] \to [p, \not\in, Y, a, \$]$, если $(p, Y, L) \in \delta(q, X)$;
(2.3) $[\not\in, q, X, a, \$] \to [\not\in, Y, a, p, \$]$, если $(p, Y, R) \in \delta(q, X)$;
(2.4) $[\not\in, X, a, q, \$] \to [\not\in, p, X, a, \$]$, если $(p, \$, L) \in \delta(q, \$)$;

} Моделируют движения на односимвольной цепочке при $q \in Q \setminus F$.

(3.1) $[q, \not\in, X, a, \$] \to a$;
(3.2) $[\not\in, q, X, a, \$] \to a$;
(3.3) $[\not\in, X, a, q, \$] \to a$;

} Восстанавливают входную односимвольную цепочку при $q \in F$.

(4.1) $A_1 \to [q_0, \not\in, a, a] A_2$;
(4.2) $A_2 \to [a, a] A_2$;
(4.3) $A_2 \to [a, a, \$]$;

} Моделируют начальные конфигурации вида $(q_0, \not\in w\$, 1)$, где $|w| > 1$.

(5.1) $[q, \not\in, X, a] \to [\not\in, p, X, a]$, если $(p, \not\in, R) \in \delta(q, \not\in)$;
(5.2) $[\not\in, q, X, a] \to [p, \not\in, Y, a]$, если $(p, Y, L) \in \delta(q, X)$;
(5.3) $[\not\in, q, X, a]\,[Z, b] \to [\not\in, Y, a]\,[p, Z, b]$, если $(p, Y, R) \in \delta(q, X)$;

} Моделируют движения на левом конце цепочки при $q \in Q \setminus F$.

(6.1) $[q, X, a]\,[Z, b] \to [Y, a][p, Z, b]$, если $(p, Y, R) \in \delta(q, X)$;
(6.2) $[Z, b]\,[q, X, a] \to [p, Z, b]\,[Y, a]$, если $(p, Y, L) \in \delta(q, X)$;
(6.3) $[q, X, a]\,[Z, b, \$] \to [Y, a]\,[p, Z, b, \$]$, если $(p, Y, R) \in \delta(q, X)$;

} Моделируют движения в середине цепочки при $q \in Q \setminus F$.

(7.1) $[q, X, a, \$] \to [Y, a, p, \$]$, если $(p, Y, R) \in \delta(q, X)$;
(7.2) $[X, a, q, \$] \to [p, X, a, \$]$, если $(p, \$, L) \in \delta(q, \$)$;
(7.3) $[Z, b]\,[q, X, a, \$] \to [p, Z, b]\,[Y, a, \$]$, если $(p, Y, L) \in \delta(q, X)$;

} Моделируют движения на правом конце цепочки при $q \in Q \setminus F$.

(8.1) $[q, \not\in, X, a] \to a$;
(8.2) $[\not\in, q, X, a] \to a$;

} Восстановление входного символа при переходе в состояние $q \in F$ на левом конце цепочки.

(8.3) $[q, X, a] \to a$;

} Восстановление входного символа при переходе в состояние $q \in F$ в середине цепочки.

(8.4) $[q, X, a, \$] \to a$;
(8.5) $[X, a, q, \$] \to a$;

} Восстановление входного символа при переходе в состояние $q \in F$ на правом конце цепочки.

(9.1) $a[X, b] \to ab$;
(9.2) $a[X, b, \$] \to ab$;

} Восстановление входной цепочки слева направо.

(9.3) $[X, a]b \to ab$;
(9.4) $[\not\in, X, a]b \to ab$;

} Восстановление входной цепочки справа налево.

Здесь $p \in Q$; $X, Y, Z \in \Gamma \setminus \{\not\in, \$\}$; $a, b \in \Sigma \setminus \{\not\in, \$\}$.

Очевидно, что построенная нами грамматика G — контекстно-зависима.

Индукцией по числу движений lba M нетрудно доказать, что если непустая цепочка принимается линейно ограниченным автоматом M, то она выводима в контекстно-зависимой грамматике G. И наоборот, индукцией по длине вывода можно доказать, что если цепочка выводима в контекстно-зависимой грамматике G, то она принимается линейно ограниченным автоматом M.

§ 8.3. Контекстно-зависимые языки — подкласс рекурсивных множеств

В гл. 2 было показано, что каждый контекстно-зависимый язык рекурсивен. Теперь мы покажем, что обратное утверждение неверно.

Теорема 8.3. *Существуют рекурсивные множества, которые не являются контекстно-зависимыми*.

Доказательство. Мы можем перечислять все строки из множества $\{0,1\}^*$, как в § 7.3. Пусть x_i — i-я строка. Аналогично мы можем перечислять все грамматики типа 0, у которых алфавит терминалов состоит из тех же символов 0 и 1. Поскольку имена нетерминалов не существенны, а каждая грамматика имеет конечное их число, можно предположить, что имеется не более чем счетное число нетерминалов, которые можно представить в бинарном коде как 01, 011, 0111, 01111, и т.д. Предполагается, что 01 — всегда начальный нетерминал. Кроме того, терминал 0 будем представлять как 00, а терминал 1 — при помощи кода 001. Символ $\rightarrow$ представим кодом 0011, а запятую, отделяющую одно правило грамматики от другого, — кодом 00111. Таким образом, символы алфавитов, используемые в правилах грамматики, представляются двоичными кодами: 00 — терминал 0, 001 — терминал 1 и 01^i, где $i = 1, 2, \ldots$ — нетерминалы. Состав нетерминального словаря конкретной грамматики определяется по самим правилам грамматики, в которых эти нетерминалы используются.

Заметим, что не все цепочки из 0 и 1 представляют грамматики, тем более контекстно-зависимые грамматики. Однако по заданной цепочке можно легко судить, является ли она csg. Поскольку имеется бесконечно много csg, мы можем их нумеровать некоторым разумным образом: $G_1, G_2, \ldots$.

Доказательство теоремы теперь тривиально.

Определим язык $L = \{x_i \mid x_i \notin L(G_i)\}$. Язык L — рекурсивен. Действительно, по любой данной цепочке x_i легко определить i, а затем можно сгенерировать G_i. Согласно теореме 2.2 имеется алгоритм, который определяет, находится ли цепочка x_i в языке $L(G_i)$, поскольку G_i — контекстно-зависимая грамматика. Следовательно, имеем алгоритм определения для любой цепочки x, находится ли она в языке L.

Теперь покажем, что язык L не порождается никакой контекстно-зависимой грамматикой. Предположим противное: язык L порождается csg G_j. Пусть $x_j \in L$. Поскольку $L = L(G_j)$, то $x_j \in L(G_j)$. Но по самому определению языка $x_j \notin L(G_j)$. Получаем противоречие, которое отвергает наше предположение о том, что существует csg G_j, порождающая язык L. Поскольку приведенное рассуждение справедливо для каждой csg G_j в перечислении и поскольку перечисление содержит каждую csg, мы заключаем, что язык L не является контекстно-зависимым языком. Следовательно, L — рекурсивное множество, которое не является контекстно-зависимым. Что и требовалось доказать.