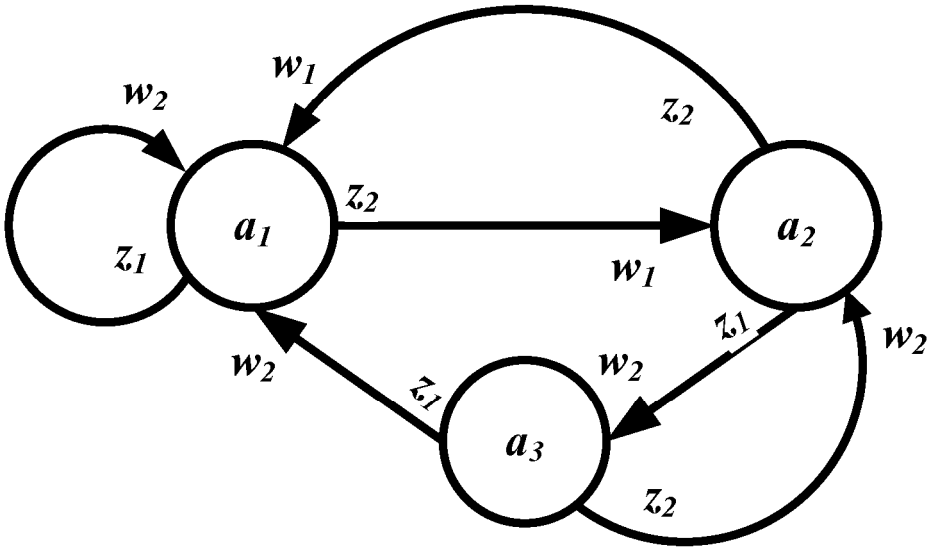
ТЕОРИЯ АВТОМАТОВ

Лекционный материал, разработанный
для самостоятельного освоения
в условиях удаленной системы обучения

часть 10



Теория сложности алгоритмов часть II

Классы $\mathbb{P}$ и $\mathbb{NP}$

Поскольку детерминированные машины Тьюринга являются частным случаем недетерминированных, то следующее определение мы сформулируем для случая недетерминированных машин.

Определение. Недетерминированная машина Тьюринга $T = \langle \mathcal{A}, Q, P, q_1, q_0 \rangle$ называется *полиномиально ограниченной над алфавитом* $\mathcal{A}_0 \subseteq \mathcal{A} \setminus \{0\}$, если существует полином $p(n)$ с натуральными коэффициентами, такой, что для любого слова $w \in \mathcal{A}_0^*$ не существует конфигурации M с условием $q_1 0 w 0 \vdash_T^{p(|w|)+1} M$.

Другими словами, любое вычисление машины заканчивается не более чем за $p(|w|)$ шагов, где $|w|$ — длина входного слова.

Определение. Язык $L \subseteq \mathcal{A}_0^*$ называется *полиномиально распознаваемым на детерминированной машине Тьюринга*, если существует детерминированная полиномиально ограниченная над алфавитом $\mathcal{A}_0$ машина Тьюринга T, которая распознает язык L.

Через $\mathbb{P}$ обозначим класс всех языков, полиномиально распознаваемых на детерминированных машинах Тьюринга.

Определение. Язык $L \subseteq \mathcal{A}_0^*$ называется *полиномиально распознаваемым на недетерминированной машине Тьюринга*, если существует недетерминированная полиномиально ограниченная над алфавитом $\mathcal{A}_0$ машина Тьюринга T, которая распознает язык L.

Через $\mathbb{NP}$ обозначим класс всех языков, полиномиально распознаваемых на недетерминированных машинах Тьюринга.

Пример. Вернемся к примеру из предыдущего параграфа. Мы построили недетерминированную машину Тьюринга T, которая распознает язык $L = \{a^n \mid n$ — составное число $\}$ над алфавитом $\mathcal{A}_0 = \{a\}$.

Допустим, на вход машины T подана начальная конфигурация $q_1 0 a^n 0$, т. е. длина входного слова равна n. Оценим сверху максимально возможную длину вычисления на построенной машине с данной начальной конфигурацией.

Недетерминированная часть вычисления, т. е. цепочка преобразований:

$$q_1 0 a^n 0 \vdash^* 0 b^{k-2} q_4 b 0 a^{n-k} 0 = 0 0^{sk+t} b^{k-2-t} q_4 b 0 b^t a^{n-(s+1)k-t} 0,$$

где $s = t = 0$, очевидно осуществляется за $k + 1$ шагов.

Далее, как мы уже говорили, запускается двойная циклическая структура. В цепочке преобразований

$$00^{sk+t}b^{k-2-t}q_4b0b^ta^{n-(s+1)k-t}0 \vdash^* 00^{sk+t}q_5b^{k-1-t}0b^ta^{n-(s+1)k-t}0 \vdash$$
$$\vdash 00^{sk+t}0q_6b^{k-2-t}0b^ta^{n-(s+1)k-t}0 \vdash^* 00^{sk+t}0b^{k-2-t}0q_7b^ta^{n-(s+1)k-t}0$$

использовано ровно $2(k-t)$ шагов. Если в состоянии q_7 обозревается 0, то происходит неправильная остановка, но длина такого вычисления не является максимальной. Иначе у нашей цепочки будет продолжение

$$00^{sk+t+1}b^{k-2-t}0q_7b^ta^{n-(s+1)k-t}0 \vdash^* 00^{sk+t+1}b^{k-2-t}q_80b^{t+1}a^{n-(s+1)k-(t+1)}0,$$

в которой использовано $2t + 1$ шагов. Далее из состояния q_8 машина переходит за 1 шаг в состояние q_9 и смещается на ячейку влево. Если в состоянии q_9 обозревается b, то машина переходит за 1 шаг на следующую итерацию во внутреннем цикле. Таким образом, одна итерация внутреннего цикла осуществляется за $2(k-t)+(2t+1)+2 = 2k + 3$ шага. Внутренний счетчик принимает значения $t = 0, \ldots, k - 2$.

Если же в состоянии q_9 обозревается 0, то, значит, $t = k - 2$ и цепочка имеет продолжение

$$00^{(s+1)k-2}q_900b^{k-1}a^{n-(s+2)k+1}0 \vdash^* 00^{(s+1)k}b^{k-1}q_{11}a^{n-(s+2)k+1}0,$$

в котором насчитывается $k+1$ шагов. Если в состоянии q_{11} обозревается 0, то происходит неправильная остановка, но длина такого вычисления не является максимальной. Иначе цепочка продолжается так (здесь использован 1 шаг):

$$00^{(s+1)k}b^{k-1}q_{11}a^{n-(s+2)k+1}0 \vdash 00^{(s+1)k}b^{k-1}0q_{12}a^{n-(s+2)k}0.$$

Если в состоянии q_{12} обозревается 0, то, значит, k делит n, машина производит правильную остановку и выдает 1. Иначе машина после исполнения последних двух команд, т. е. за 2 шага, переходит на следующую итерацию во внешнем цикле.

Таким образом, внешний цикл состоит из $k - 1$ повторений внутреннего цикла и из дополнительных $k + 5$ шагов. Всего в одной итерации внешнего цикла получается $(k - 1)(2k + 3) + k + 5 = 2k^2 + 2k + 2$ шагов.

Внешний счетчик в самом худшем случае принимает значения $s = 0, \ldots, [\frac{n}{k}]$. Следовательно, учитывая шаги из стартовой части вычислений, получаем, что общее число шагов в вычислении не превосходит

$$k + 1 + (2k^2 + 2k + 2)(\left[\frac{n}{k}\right] + 1) \leqslant n + 1 + (2n^2 + 2n + 2)(n + 1) = 2n^3 + 4n^2 + 5n + 3.$$

Следовательно, длина вычислений ограничена сверху полиномом $p(n) = 2n^3 + 4n^2 + 5n + 3$ и, значит, язык L полиномиально распознаваем на недетерминированной машине Тьюринга, т. е. $L \in \mathbb{NP}$.

Теперь мы сформулируем некоторые важные свойства классов $\mathbb{P}$ и $\mathbb{NP}$.

Предложение 74. *Имеет место включение* $\mathbb{P} \subseteq \mathbb{NP}$.

Доказательство. Очевидно, поскольку детерминированные машины Тьюринга являются частным случаем недетерминированных. □

Замечание. Вопрос о справедливости обратного включения $\mathbb{NP} \subseteq \mathbb{P}$ является одной из важнейших нерешенных проблем современной математики, которую очень часто обозначают как *ПРОБЛЕМА* «$\mathbb{P} = \mathbb{NP}$?».

Для машин Тьюринга справедлива теорема о детерминизации, т. е. *для любой недетерминированной машины Тьюринга, распознающей некоторый язык L, существует детерминированная машина Тьюринга, которая распознает тот же самый язык L.* Доказательство данной теоремы можно найти в [12]. Таким образом, класс всех языков, распознаваемых недетерминированными машинами Тьюринга, совпадает с классом всех языков, распознаваемых детерминированными машинами Тьюринга, но данный факт никак не решает проблему «$\mathbb{P} = \mathbb{NP}$?». Причина этого заключается в том, что процедура детерминизации недетерминированной машины Тьюринга очень сильно увеличивает временную сложность машины. Наименьшая верхняя граница сложности, которая возникает после применения процедуры детерминизации, экспоненциальная.

Предложение 75. *Класс $\mathbb{P}$ замкнут относительно дополнения.*

Доказательство. Пусть язык $L \subseteq \mathcal{A}_0^*$ принадлежит классу $\mathbb{P}$. Следовательно, найдется детерминированная полиномиально ограниченная над алфавитом $\mathcal{A}_0 \subseteq \mathcal{A} \setminus \{0\}$ машина Тьюринга $T = \langle \mathcal{A}, Q, P, q_1, q_0 \rangle$, которая распознает язык L.

Определим машину Тьюринга $T' = \langle \mathcal{A}, Q', P', q_1, q_0' \rangle$, где $Q' = Q \cup \{q_0'\}$, $P' = P \cup \{q_0 0 \to q_0' 1, q_0 1 \to q_0' 0, \dots\}$. Другими словами, мы добавили в машину T новое заключительное состояние q_0' и две команды $q_0 0 \to q_0' 1$ и $q_0 1 \to q_0' 0$ (конкретный вид команд $q_0 a_i \to \dots$, где $a_i \notin \{0, 1\}$, не имеет значения).

Ясно, что определенная таким образом машина T' полиномиально ограничена над алфавитом $\mathcal{A}_0$ и распознает дополнение $\mathcal{A}_0^* \setminus L$. □

Замечание. Отметим без доказательства, что класс $\mathbb{P}$ замкнут также относительно объединения, пересечения, конкатенации и звездочки Клини.

Класс $\mathbb{NP}$ также замкнут относительно объединения, пересечения, конкатенации и звездочки Клини. Однако вопрос о замкнутости класса $\mathbb{NP}$ относительно дополнения до сих пор является открытым и тесно связан с проблемой «$\mathbb{P} = \mathbb{NP}$?».

Предложение 76. *Если класс $\mathbb{NP}$ не замкнут относительно операции дополнения, то $\mathbb{P} \neq \mathbb{NP}$.*

Доказательство. Допустим, существует язык L такой, что $L \in \mathbb{NP}$, но его дополнение $\overline{L} \notin \mathbb{NP}$. Докажем, что в этом случае $L \notin \mathbb{P}$. Если, напротив, $L \in \mathbb{P}$, то в силу предложения 75 $\overline{L} \in \mathbb{P}$. Следовательно, в силу предложения 74 заключаем, что $\overline{L} \in \mathbb{NP}$, что противоречит нашему первоначальному предположению. □

В заключение параграфа докажем, что класс $\mathbb{P}$ нетривиален, т. е. является собственным подклассом в классе всех языков, распознаваемых (детерминированными или недетерминированными) машинами Тьюринга.

Теорема 77. *Существует распознаваемый машиной Тьюринга язык L, не принадлежащий классу $\mathbb{P}$.*

Доказательство. Для определения требуемого языка L нам необходимо закодировать каждую машину Тьюринга в виде слова над некоторым алфавитом. Пусть

$c = q_i a_j \to q_k a_l S$, где $S \in \{\Lambda, R, L\}$, — произвольная команда машины Тьюринга. Сопоставим ей слово

$$\widehat{c} = 1^i \mid 1^j \mid 1^k \mid 1^l \mid 1^s,$$

где $s = 0$, если $S = \Lambda$; $s = 1$, если $S = R$; и $s = 2$, если $S = L$. Если теперь T — произвольная детерминированная машина Тьюринга с программой $\{c_1, \ldots, c_m\}$, то сопоставим ей слово

$$\widehat{T} = \widehat{c_1} \parallel \widehat{c_2} \parallel \ldots \parallel \widehat{c_m}.$$

Зафиксируем алфавит $\mathcal{A}_0 = \{1, |, \|\}$ и рассмотрим над этим алфавитом следующий язык:

$$\begin{aligned} L = \{\widehat{T} \mid &\text{машина Тьюринга } T \text{ перерабатывает конфигурацию} \\ &q_1 0 \widehat{T} 0 \text{ в конфигурацию } u q_0 1 v \text{ для некоторых } u \text{ и } v, \\ &\text{не более, чем за } 2^{|\widehat{T}|} \text{ шагов}\}. \end{aligned}$$

Язык L распознаваем машиной Тьюринга. Действительно, существует интуитивно вычислимая процедура (машина Тьюринга), которая по произвольному слову w над алфавитом $\mathcal{A}_0$ сначала проверяет, является ли оно словом вида $\widehat{T}$ для какой-либо детерминированной машины T. Затем, если w имеет вид $\widehat{T}$, данная процедура восстанавливает программу T и проверяет, действительно ли она перерабатывает конфигурацию $q_1 0 \widehat{T} 0$ в конфигурацию вида $u q_0 1 v$ не более, чем за $2^{|\widehat{T}|}$ шагов. Описанную процедуру можно смоделировать на машине Тьюринга.

Допустим теперь, что $L \in \mathbb{P}$. Следовательно, по предложению 75 язык $\overline{L}$ полиномиально распознается на некоторой детерминированной машине Тьюринга M. Пусть $p(x)$ — соответствующий полином из определения полиномиальной ограниченности M. Без ограничения общности можно считать, что

$$p(|\widehat{M}|) < 2^{|\widehat{M}|}.$$

Действительно, поскольку $\frac{p(n)}{2^n} \to 0$ при $n \to \infty$, найдется $n_0 \in \omega$ такое, что для всех $n \geqslant n_0$ выполняется $p(n) < 2^n$. Отсюда следует, что, добавив в машину M необходимое число фиктивных состояний и команд (т. е. команд, не влияющих на работу машины), мы можем увеличить число $|\widehat{M}|$ так, чтобы имело место неравенство $|\widehat{M}| \geqslant n_0$, из которого вытекает справедливость условия $p(|\widehat{M}|) < 2^{|\widehat{M}|}$.

Исследуем теперь вопрос принадлежности слова $\widehat{M}$ языку L. Условие $\widehat{M} \in L$ эквивалентно следующему условию:

1) *Машина M перерабатывает конфигурацию $q_1 0 \widehat{M} 0$ в конфигурацию $u q_0 1 v$ для некоторых u и v не более, чем за $2^{|\widehat{M}|}$ шагов.*

Поскольку машина M, будучи запущенная из конфигурации $q_1 0 \widehat{M} 0$, всегда останавливается не более, чем за $p(|\widehat{M}|) < 2^{|\widehat{M}|}$ шагов, то условие (1) равносильно следующему условию:

2) *Машина M перерабатывает конфигурацию $q_1 0 \widehat{M} 0$ в конфигурацию вида $u q_0 1 v$.*

Наконец, вспомним, что машина M распознает язык $\overline{L}$. Отсюда вытекает, что условие (2) эквивалентно условию $\widehat{M} \notin L$.

Таким образом, имеет место эквивалентность $\widehat{M} \in L \iff \widehat{M} \notin L$. Противоречие. Следовательно, $L \notin \mathbb{P}$. $\square$

$\mathbb{NP}$-полные проблемы

Еще никому не удалось доказать, что существует язык из класса $\mathbb{NP}$, не принадлежащий классу $\mathbb{P}$. Проблема «$\mathbb{P} = \mathbb{NP}$?» по-прежнему остается открытой. Однако можно доказать, что некоторые языки не менее «трудны», чем любой язык из $\mathbb{NP}$, в том смысле, что если бы у нас был детерминированный алгоритм, распознающий один из этих «не менее трудных» языков за полиномиальное время, то можно было бы для каждого языка из класса $\mathbb{NP}$ найти детерминированный алгоритм, распознающий его за полиномиальное время. Такие языки называются $\mathbb{NP}$-*полными*.

Для определения $\mathbb{NP}$-полноты необходимо сначала ввести понятие полиномиальной сводимости, которое в свою очередь использует понятие полиномиально вычислимой функции.

Определение. Пусть $T = \langle \mathcal{A}, Q, P, q_1, q_0 \rangle$ — детерминированная машина Тьюринга и задан некоторый алфавит $\mathcal{A}_0 \subseteq \mathcal{A} \setminus \{0\}$. Говорят, что машина T *вычисляет* всюду определенную словарную функцию $f : \mathcal{A}_0^* \to \mathcal{A}_0^*$, если для любого слова $w \in \mathcal{A}_0^*$ имеет место

$$q_1 0 w 0 \underset{T}{\Longrightarrow} q_0 0 f(w) 0 0^s, \text{ для некоторого } s \geqslant 0.$$

Другими словами, машина T на входе $q_1 0 w 0$ всегда останавливается, преобразуя начальную конфигурацию без достраивания новых ячеек слева в конфигурацию $q_0 0 f(w) 00 \ldots 0$.

Функция $f : \mathcal{A}_0^* \to \mathcal{A}_0^*$ называется *полиномиально вычислимой*, если существует полиномиально ограниченная над алфавитом $\mathcal{A}_0$ детерминированная машина Тьюринга, которая вычисляет f.

Замечание. Для всюду определенных функций вида $f : \omega \to \omega$ предыдущее определение согласуется со старым определением вычислимой по Тьюрингу частичной функции. При этом мы, как обычно, заменяем числовую функцию $f : \omega \to \omega$ на ее словарный аналог $f : \{1\}^* \to \{1\}^*$ по правилу $f(1^n) = 1^{f(n)}$ и рассматриваем машины Тьюринга над алфавитом $\mathcal{A} = \{0, 1\}$.

Таким образом, любая полиномиально вычислимая функция $f : \omega \to \omega$ обязана быть частично вычислимой. Возникает естественный вопрос: существуют ли частично вычислимые функции, которые не вычислимы полиномиально? Интуитивные рассуждения, приведенные во введении в данную главу, дают нам основания для утвердительного ответа на этот вопрос. Ниже мы дадим более формальное доказательство этого факта.

Теорема 78. *Существует двухместная вычислимая функция $F(n, x)$, универсальная для семейства $\{f : \omega \to \omega \mid f$ — полиномиально вычислима$\}$.*

Доказательство. Мы изложим только идею доказательства. Пусть $p_m(x)$ — универсальная вычислимая функция для семейства всех полиномов с натуральными коэффициентами (см. предложение 65).

Неформальный алгоритм для вычисления функции $F(n, x)$ имеет следующее описание. Запускаем машину Тьюринга с номером $(n)_0$ на входной конфигурации $q_1 01^x 0$ и проделываем ровно $p_{(n)_1}(x)$ шагов работы машины. Если за данное количество шагов вычисление успешно завершилось, т. е. машина перешла без достраивания новых ячеек слева в конфигурацию вида $q_0 01^y 0 \ldots 0$, то выдаем y в качестве значения $F(n, x)$. В противном случае выдаем 0.

Определенная подобным образом функция F является искомой. □

Следствие 79. *Существует вычислимая функция, не являющаяся полиномиально вычислимой.*

Доказательство. Допустим, напротив, любая вычислимая функция вида $f : \omega \to \omega$ полиномиально вычислима. Следовательно, семейство всех одноместных вычислимых функций совпадает с семейством $\{f : \omega \to \omega \mid f$ — полиномиально вычислима$\}$. Тогда вычислимая функция $F(n, x)$, построенная в теореме 78, будет универсальной для семейства всех одноместных вычислимых функций. Последнее противоречит предложению 62. □

Определение. Говорят, что язык $L_1 \subseteq \mathcal{A}_0^*$ *полиномиально сводится* к языку $L_2 \subseteq \mathcal{A}_0^*$, если существует полиномиально вычислимая словарная функция $f : \mathcal{A}_0^* \to \mathcal{A}_0^*$, такая, что для любого слова $w \in \mathcal{A}_0^*$ выполняется:

$$w \in L_1 \iff f(w) \in L_2.$$

В теории сложности традиционно языки отождествляют с проблемами вхождения слов в эти языки. В данной терминологии полиномиальная сводимость языка L_1 к языку L_2 означает, что проблема L_1 не сложнее, чем проблема L_2. Если существует детерминированная машина, разрешающая проблему L_2 за полиномиальное время, то ее можно эффективно преобразовать в детерминированную машину, которая разрешает проблему L_2 также за полиномиальное время.

Определение. Язык $L_0 \subseteq \mathcal{A}_0^*$ называется $\mathbb{NP}$-*полным*, если $L_0 \in \mathbb{NP}$ и любой язык $L \in \mathbb{NP}$ полиномиально сводится к L_0.

Из определения следует, что все $\mathbb{NP}$-полные языки сводятся друг к другу. Семейство $\mathbb{NP}$-полных языков играет очень важную роль в теории сложности. Если хотя бы для одной из $\mathbb{NP}$-полных проблем удастся найти детерминированный алгоритм, решающий эту проблему за полиномиальное время, то будет справедливо равенство $\mathbb{P} = \mathbb{NP}$, и все остальные проблемы из этого семейства также будут решаться с помощью детерминированных полиномиальных алгоритмов.

Теорема 80. *Пусть L — произвольный $\mathbb{NP}$-полный язык. Равенство $\mathbb{P} = \mathbb{NP}$ выполняется тогда и только тогда, когда $L \in \mathbb{P}$.*

Доказательство. Докажем сначала необходимость. Пусть $\mathbb{P} = \mathbb{NP}$. Из определения $\mathbb{NP}$-полноты вытекает, что $L \in \mathbb{NP}$. Следовательно, $L \in \mathbb{P}$.

Теперь докажем достаточность. Пусть $L \in \mathbb{P}$. Следовательно, найдется детерминированная машина Тьюринга T_1, полиномиально ограниченная полиномом $p_1(n)$, которая распознает L. Пусть далее L' — произвольный язык из класса $\mathbb{NP}$. Требуется показать, что $L' \in \mathbb{P}$.

В силу $\mathbb{NP}$-полноты языка L язык L' полиномиально сводится к L посредством некоторой полиномиально вычислимой функции f. Следовательно, существует детерминированная машина Тьюринга T_2, полиномиально ограниченная полиномом $p_2(n)$, которая вычисляет f.

Рассмотрим композицию машин T_2T_1. Очевидно, что T_2T_1 является детерминированной машиной. Машина T_2T_1 распознает слово w тогда и только тогда, когда $f(w) \in L$. Условие $f(w) \in L$, в свою очередь, эквивалентно условию $w \in L'$. Таким образом, T_2T_1 в точности распознает язык L'.

Докажем, наконец, что машина T_2T_1 полиномиально ограничена. Работа машины T_2T_1 на произвольном входном слове w состоит из двух этапов. На первом этапе машина T_2 перерабатывает w в слово $f(w)$ за не более, чем $p_2(|w|)$ шагов. Поскольку машина T_2 за один такт работы способна записать на ленту не более одного символа, то длина слова $f(w)$ не превосходит $p_2(|w|) + |w|$. Следовательно, на втором этапе машина T_1 перерабатывает слово $f(w)$ в определенное выходное слово за не более, чем $p_1(p_2(|w|) + |w|)$ шагов. Суммируя все подсчеты, заключаем, что машина T_2T_1 останавливается на входе w за не более, чем $p_2(|w|) + p_1(p_2(|w|) + |w|)$ шагов, что является полиномом от $|w|$. $\square$