ЛИТА ДИСКРА РК1

Теория

Определения

Машина Тьюринга

Машина Тьюринга определяется кортежем вида

$$T = (Q, V, *, \square, S, L, R, q_0, q_f, \delta),$$

где

- Q - конечное множество состояний
- V - конечный входной алфавит
- $* \notin V$ - маркер начала ленты
- $\square \notin V$ - пробел
- $S, L, R \notin V$ - направления движения головки
- $q_0 \in Q$ - начальное состояние
- $q_f \in Q$ - заключительное состояние
- δ - функция переходов (по сути программа):
  - $\delta : Q \times (V \cup \{*, \square\}) \to 2^{\{Q \times (V \cup \{*, \square\}) \times \{S, L, R\}\}}$

Конфигурация МТ и отношения выводимости на множестве конфигураций

Конфигурация машина Тьюринга T есть кортеж

$$(q, x, y) \in Q \times (V \cup \{*, \square\})^* \times (V \cup \{*, \square\})^*$$

Из конфигурации $C = (q, x, ay)$ непосредственно выводится $C' = (r, x, by)$,
если $qa \to rb, S \in \delta$
Из конфигурации $C = (q, xc, ay)$ непосредственно выводится $C' = (r, x, cby)$,
если $qa \to rb, L \in \delta$
Из конфигурации $C = (q, x, acy)$ непосредственно выводится $C' = (r, xb, cy)$,
если $qa \to rb, R \in \delta$

Вычислимость по Тьюрингу

МТ применима к слову x: $!T(x) \Longleftrightarrow (q_0, \lambda, *x\square) \vdash^* (q_f, \lambda * y\square);\ y \Longleftrightarrow T(x)$ - результат работы МТ со словом x

Вербальная функция $f : V^* \underset{\cdot}{\longrightarrow} V^*$ (то есть частичная функция типа $(V,\ V)$) называется вычислимой по Тьюрингу, если может быть построена такая МТ T_f на алфавитом V, что для любого слова $x \in V^*$ выполняется:
$!T_f(x) \Leftrightarrow (x \in D(f))\ \&\ (T_f(x) = f(x))$

Нормальный алгорифм Маркова

Нормальный алгорифм в алфавите V есть упорядоченная тройка $A = (V, S, P)$, где S - схема НА, представляющая собой упорядоченный набор формул подстановки в алфавите V, в котором отмечены некоторые формулы, называемые заключительными и образующие частичный кортеж P.

Процесс работы НА со словом

Процесс работы НА A со словом x есть конечная или бесконечная последовательность слов $x = x_0, x_1 ... x_n ...$, где $n \geq 0$, причем для каждого $i \geq 0 \;\; A : x_i \vdash x_{i+1}$ или $A : x_i \vdash \cdot \; x_{i+1}$, если слово x_{i+1} определено в последовательности. Это слово, как и все последующие, считается неопределенным, если $A : x_{i-1} \vdash \cdot \; x_i$, то есть последнее слово получено применением заключительной формулы, или ~ $A(x_i)$, то есть последнее слово не поддается НА и имеет место естественный обрыв процесса работы.

Если процесс работы НА A со словом x конечен, то его последнее слово обозначается $A(x)$ и называется **результатом работы НА** A со словом x. В этом случае говорят, что НА применим к слову и пишут $!A(x)$, иначе не применим и пишут $\neg !A(x)$

Вычислимость по Маркову

Вербальная функция f типа $(V, \; V)$ называется вычислимой по Маркову, если может быть построен НА A_f над алфавитом V такой, что для любого слова $x \in V^*$ имеет место следующее: $!A_f(x)$ тогда и только тогда, когда $x \in D(f)$ и в этом случае $A_f(x) = f(x)$

Большая теория

Определения изображения и записи НА. Примеры. Формулировка теоремы об универсальном НА

(пусто)

Доказать алгоритмическую неразрешимость проблемы применимости для НА

(пусто)

Понятие МП-автомата. Конфигурации, выводимость на множестве конфигураций. Язык, допускаемый МП-автоматом. Построение МП-автомата по КС-грамматике

(пусто)

Теоремы объединения, разветвления и повторения НА (формулировки). Построение НА, распознающего равенство слов

(пусто)

Проблемы применимости и самоприменимости для НА. Доказательство неразрешимости проблемы самоприменимости

(пусто)

Определение разрешимого и перечислимого языка. Связь разрешимости и перечислимости. Примеры. Доказать невозможность разрешающего НА для языка, для которого невозможен полуразрешающий НА

(пусто)

Понятие перевода в двухбуквенный алфавит. Формулировка теоремы о переходе

Пусть $V = \{a_1, ...a_n\};\ V_\alpha = \alpha, \beta$. Тогда перевод буквы $a_k \in V$ в V_α слово: $a_k = \alpha\beta^k\alpha$.

Каким бы ни был НА A типа (V, V) над V, может быть построен НА B в алфавите $V \cup V_\alpha$, вполне эквивалентный A относительно V, т.е. $\forall x \in V^* : A(x) \simeq B(x)$

Практика

МТ (Машина Тьюринга)

МТ, аннулирующую все слова в алфавите {a, b}, которые содержат вхождение слова aaba

$$q_0 * \to q_0 *,\ R$$
$$q_0 a \to q_1 a,\ R \quad //a$$
$$q_0 b \to q_0 b,\ R$$
$$q_1 a \to q_2 a,\ R \quad //aa$$
$$q_1 b \to q_0 b,\ R$$
$$q_2 a \to q_2 a,\ R$$
$$q_2 b \to q_3 b,\ R \quad //aab$$
$$q_3 a \to q_4 a,\ R \quad //aaba!$$
$$q_3 b \to q_0 b,\ R$$
$$q_4 p \to q_4 p,\ R$$
$$q_4 \square \to q_5 \square,\ L$$
$$q_5 p \to q_5 \square,\ L$$
$$q_5 * \to q_f *,\ S$$
$$q_i \square \to q_6 \square,\ L,\ i = 0, 1, 2, 3$$
$$q_6 p \to q_6 p,\ L$$
$$q_6 * \to q_f *,\ S$$

МТ, аннулирующую все слова в алфавите {a, b}, которые содержат вхождение слова aaab

(по аналогии с первой)

МТ, аннулирующую все слова в алфавите {a, b, c}, которые содержат вхождения слов aab и cab

Слова не накладываются друг на друга, значит будет как минимум 4 особых состояния: когда ни одно слово не найдено, когда найдено ааб и не найдено саб, когда найдено саб и не найдено ааб и когда оба слова найдены.

МТ, аннулирующую все слова в алфавите {a, b}, которые содержат не менее 2-х (двух) вхождений слова aaab

Сначала мы ищем первое вхождение слова, если нашли, то следующее вхождение мы будем помнить, что оно второе, по состояниям. Нужно также не забывать в конце обработать случай, когда после всех неучтеннных окончаний пишется пробел и что тогда происходит

МТ, аннулирующую все слова в алфавите {a, b}, которые содержат ровно одно вхождение слова aaab

$$q_0 * \to q_0 *,\ R q_0 a \to q_1 a,\ R \quad //a$$
$$q_0 b \to q_0 b,\ R \quad //b$$
$$q_1 a \to q_2 a,\ R \quad //aa$$
$$q_1 b \to q_0 b,\ R \quad //ab$$
$$q_2 a \to q_3 a,\ R \quad //aaa$$

$q_2 b \to q_3 a,\ R \quad //aaa..a$
$q_3 b \to q_4 b,\ R \quad //aaab!$
$q_4 a \to p_1 a,\ R \quad //$"ищем" второе aaab
$q_4 b \to q_4 b,\ R$
$p_1 a \to p_2 a,\ R \quad //aa$
$p_1 b \to q_4 b,\ R$
$p_2 a \to p_3 a,\ R \quad //aaa$
$p_3 a \to p_3 a,\ R$
$p_3 b \to p_4 b,\ S \quad //$увы, ещё одно aaab
$S\square \to q_5\square,\ L \quad //S \in \{q_4, p_1, p_2, p_3\}$
$q_5 \xi \to q_5 \square,\ L$
$q_5 * \to q_f *,\ S$
$p_4 \xi \to p_4 \xi,\ R$
$R\square \to q_6\square,\ L \quad //R \in \{q_0, q_1, q_2, q_3, p_4\}$
$q_6 \xi \to q_6 \xi,\ L$
$q_6 * \to q_f *,\ S$

МТ, аннулирующую все слова в алфавите {a, b}, которые содержат не менее 2-х (двух) вхождений слова aba

$q_0 * \to q_0 *,\ R$
$q_0 \square \to q_f \#,\ S$
$q_0 a \to q_a a,\ R$
$q_0 b \to q_0 b,\ R$
$q_a a \to q_a a,\ R$
$q_a b \to q_{ab}\ b,\ R$
$q_{ab}\ a \to q_1 a,\ S$
$q_{ab}\ b \to q_0 b,\ R$
$q_1 a \to q_{1a}\ a,\ R$
$q_1 b \to q_1 b,\ R$
$q_{1a}\ a \to q_{1a}\ a,\ R$
$q_{1a}\ b \to q_{1ab}\ b,\ R$
$q_{1ab}\ a \to q_{win}\ a,\ R$
$q_{1ab}\ b \to q_1 b,\ R$
$q_{win}\ \alpha \to q_{win}\ \alpha,$
$q_{win}\ \square \to q_{del}\ \square,\ L$
$q_{del}\ \alpha \to q_{del}\ \square,\ L$
$q_{del} * \to q_f *,\ S$
$R\ \square \to q_{lose}\ \square,\ L \quad //R \notin \{q_{win},\ q_{del}\}$
$q_{lose}\ \alpha \to q_{lose}\ \alpha,\ L$
$q_{lose} * \to q_f *,\ S$

Яблоки (ну не сливы же 2025 года)

МТ, аннулирующая слово, где есть вхождение aaba, но нет вхождения baba

ровно одно вхождение aabaaa - повторилось

есть вхождение aab и aba

ровно одно вхождение aabaa

не менее двух вхождений abca

ровно одно вхождение ababa

ровно одно вхождение baba

не менее трех вхождений aaab

ровно два вхождения aaba

ровно одно вхождение aaba - повторилось

есть вхождение aaba, но нет bb

НА (Нормальный алгорифм Маркова)

НА, который аннулирует все в алфавите {a, b}, содержащие не менее 2-х (двух) вхождений слова aba

Алгоритм для решения задачи "не менее n вхождений" везде одинаков. U = aba

$$
\begin{cases}
\#\#\xi \to \xi\#\# \\
\#\# \to \$ \\
\xi\$ \to \$ \\
\$ \to \cdot \\
\#U \to U(1)\#\#U(2)U(3)...U(k) \\
\#\xi \to \xi\# \\
\# \to \cdot \\
U \to U(1)\#U(2)U(3)...U(k) \\
\xi \to \cdot\xi \\
\to \cdot
\end{cases}
$$

НА, который аннулирует все в алфавите {a, b}, содержащие не менее 3-х (трех) вхождений слова aaba

НА, который аннулирует все в алфавите {a, b}, содержащие не менее 3-х (трех) вхождений слова aaab

Добавляем к предыдущей проге обработку трех решеток и не забываем банить случай с двумя решетками, потому что этого мало

НА, который аннулирует все в алфавите {a, b}, содержащие ровно 2 (два) вхождения слова aaba

К предыдущей программе в самое начало буквально добавляется команда $\#\#U \to \cdot U$, что буквально отрубает бОльшее количество вхождений слова.

НА, который аннулирует все в алфавите {a, b}, содержащие ровно одно вхождения слова aaba

НА, который аннулирует все в алфавите {a, b}, содержащие ровно одно вхождения слова aba

НА, который аннулирует все слова в алфавите {a, b}, содержащие не менее двух вхождений baba, но не содержат вхождения abaa

Пусть $U = baba,\ V = abaa$. С самого начала просто чекать, что нет вхождения V (если один раз сработало, то и в остальные разы тоже сработает, потому что мы ли разделяем решетками, или убираем решетки, или удаляем с конца, и ничто из этого не провоцирует регулярку V вдруг заработать)

$$
\begin{cases}
V \to \cdot V \\
\#\#\xi \to \xi\#\# \\
\#\# \to \$ \\
\xi\$ \to \$ \\
\$ \to \cdot \\
\#U \to U(1)\#\#U(2)U(3)...U(k) \\
\#\xi \to \xi\# \\
\# \to \cdot \\
U \to U(1)\#U(2)U(3)...U(k) \\
\xi \to \cdot\xi \\
\to \cdot
\end{cases}
$$

Яблоки (ну не сливы же 2025 года)

НА, который аннулирует все слова, где не менее 3-х вхождений aaab

не менее двух вхождений bab и ни одного aba

не менее трех вхождений abba

не менее трех вхождений abab, не содержит baaa

не менее двух вхождений слова u, без вхождений v. U и V накладываются в общем

не менее двух вхождений acb, не содержит cbc

ровно одно вхождение u, без вхождений v

ровно одно вхождение aabaa

не менее трех вхождений aaba

ровно три вхождения baba

ровно одно вхождение abab

не менее трех вхождений abab

не менее трех aaab и не содержит aaaa