Алгоритм Маркова

О. **Алфавитом** называется всякое непустое множество символов, а сами символы алфавита называются буквами.

Пример алфавита.

Конечное множество символов {a,+,?,v}.

О. **Словом** в данном алфавите А называется всякая конечная последовательность букв алфавита А.

О. Пустая последовательность букв называется **пустым словом** и обозначается через **Λ** (или ∅).

Для любого слова Р имеем

РΛ =Λ Р=Р.

О. Под **алгоритмом в алфавите А** понимается алгоритм, входами и выходами которого являются слова в алфавите А.

О. Пусть Р слово в алфавите А. Говорят, что алгоритм 𝔄 применим к слову Р, если применение его к слову Р приводит через конечное число шагов к некоторому слову Q. При этом слово Q обозначается через 𝔄(Р).

Если процесс переработки (преобразования) слова Р бесконечен, то считается, что алгоритм ***не применим*** к этому слову.

О. ***Нормальный алгоритм над алфавитом А*** **отличается от** ***нормального алгоритма в алфавите А*** только тем, что в словах P_i

и Q_i ($1 \le i \le n$) могут использоваться не только буквы из алфавита А, но и буквы, не принадлежащие алфавиту А.

О. Формула подстановки **P→Q** называется **простой подстановкой** и означает, что **вместо P нужно подставить слово Q и перейти к следующей подстановке**.

О. Формула подстановки P→ • **Q** называется **заключительной подстановкой** и означает, что **вместо P нужно подставить Q и закончить процесс** преобразования.

Пусть P→ (•)Q означает любую из формул подстановки

(P→ Q или P→•Q).

О. **Нормальный алгоритм** в алфавите А считается **заданным**, если задана конечная таблица (схема) формул подстановок слов алфавита А:

$$B = \begin{cases} P_1 \to (\bullet) Q_1 \\ P_2 \to (\bullet) Q_2 \\ \cdots\cdots\cdots\cdots \\ P_n \to (\bullet) Q_n \end{cases}$$

Пример нормального алгоритма

1. Пусть А={*a*,*b*,*c*}. Таблица формул подстановок

$$\boldsymbol{B}= \begin{cases} a \to a & (1) \\ cc \to (\bullet)c & (2) \\ b \to c & (3) \end{cases}$$

задает некоторый нормальный алгоритм $\mathcal{B}$ в алфавите А.

Если взять слово R_o=bb, то $\mathcal{B}$ преобразует его сначала с помощью подстановки (3) в слово cb, затем, вновь применяя подстановку (3) получим cc.

Далее, по заключительной подстановке (2), получим c.

Это слово и будет $\mathcal{B}(R_o)$.

Если исходное слово R_o будет содержать букву *a*, то $\mathcal{B}$ не применим к R_o, ибо подстановка (1) будет применяться безостановочно.

Пример.

Пусть A={*a, b, c*}. Рассмотрим таблицу формул подстановок

$$\mathcal{B}=\begin{cases} \beta a \rightarrow a\beta & (1) \\ \beta b \rightarrow b\beta & (2) \\ \beta c \rightarrow c\beta & (3) \\ \beta \rightarrow \bullet a & (4) \\ \Lambda \rightarrow \beta & (5) \end{cases}$$

Таблица задает нормальный алгоритм над алфавитом А, ибо $\beta \notin A$.

Если взять произвольное слово R_o в А, то к нему сначала подстановки (1), (2) и (3) не применимы, так как слов βa, βb и βc в слове R_o быть не может.

Подставив вместо самого левого пустого слова (Λ) в R_o слово β по (5), имеем βR_o.

Если первой в R_o стоит буква *a*, то применяем подстановку (1),

если же первая буква - *b*, то применяем подстановку (2),

а если первая *c*, то применяем (3) и перемещаем β за первую букву в слове R_o.

Повторяя этот процесс столько раз, сколько букв в слове R_o, получим $R_o\beta$.

По подстановке (4) окончательно имеем $R_o a$, т.е. этот алгоритм приписывает к произвольному слову R_o в алфавите А справа от R_o букву *a*.

$$B=\begin{cases} \beta a \rightarrow a\beta & (1) \\ \beta b \rightarrow b\beta & (2) \\ \beta c \rightarrow c\beta & (3) \\ \beta \rightarrow \bullet a & (4) \\ \Lambda \rightarrow \beta & (5) \end{cases}$$

Если P=abc, то

(5)→(1) → (2) → (3) →(4)

abc => **Ø**abc → **β**abc → **aβ**bc → **abβ**c → **abcβ** →**abca**

Вариант 1

Задание.

Постройте нормальный алгоритм для преобразования слова P в слово Q, при условии, что в каждой подстановке R—((•)S число букв в словах R и S не должно превышать 3.

P=*aabcc*, Q=*aabccdabcdd*

Решение (1)

Q=*aabcc*|*da*|*bc*|*dd*; A={*a*, *b*, *c*, *d*}; $x \in$ A; {α, β, γ} $\notin$ A

$$B = \begin{cases} \alpha\, x \to x\, \alpha & (1) \\ \alpha \to d\, a\, \beta & (2) \\ \beta \to b\, c\, \gamma & (3) \\ \gamma \to \bullet\, d\, d & (4) \\ \varnothing \to \alpha & (5) \end{cases} \quad \text{или} \quad \tilde{B} = \begin{cases} \alpha\beta\, x \to x\, \alpha\beta \\ \alpha \to d\, a\, b \\ \beta \to \bullet\, c\, d\, d \\ \varnothing \to \alpha\beta \end{cases}$$

aabcc = ∅*aabcc* →α*aabcc* *(5)* →

→{*a*α*abcc*, *aa*α*bcc*, *aab*α*cc*, *aabc*α*c*, *aabcc*α *(1)*} →

→ *aabccda*β *(2)* → *aabccdabc*γ *(3)* → *aabccdabcdd (4)*

$$\alpha x \to x\alpha \quad \text{или} \quad \begin{cases} \alpha a \to a\, \alpha \\ \alpha b \to b\, \alpha \\ \alpha c \to c\, \alpha \\ \alpha d \to d\, \alpha \\ \ldots \end{cases}$$

Алгоритм Тьюринга

Рис.

Конечное множество символов $S_0,S_1,..,S_n$, которое называется алфавитом машины. Пусть S_0 означает пустой квадрат

Машина обладает некоторым конечным множеством внутренних состояний $\{q_0,q_1,...,q_m\}$.

Команды:

$$1)\ q_j\ S_i\ S_k\ q_r;$$
$$2)\ q_j\ S_i\ L\ q_r;$$
$$3)\ q_j\ S_i\ R\ q_r.$$

Пример.

1. Машина Т, заданная командами

$$q_0aRq_0$$
$$q_0bRq_0$$
$$q_0S_0aq_1,$$

Приписывает к любому слову Р алфавита $\{a,b\}$ справа букву a и останавливается. (Р=ab, Q=aba). S_0 – пусто.

было	***a***	***b***	S_0	S_0	S_0	S_0	S_0	S_0
стало	***a***	***b***	***a***	S_0	S_0	S_0	S_0	S_0
	q_0	q_0	$q_0 \to q_1$					

♦

$q_0 \to q_1$ (состояние q_0 изменили на состояние q_1 , т.е. провели переадресацию).

Задание.

Постройте машину Тьюринга для преобразования слова P в слово Q.

Пример.

P=aabc, Q=aabccdab

Решение (2)

Q=aabc|cdab; S={S_0, a, b, c, d}

1) $q_0 a\ R q_0$
2) $q_0 b\ R q_0$
3) $q_0 c\ R q_0$
4) $q_0 S_0\ c q_1$
5) $q_1 c\ R q_1$
6) $q_1 S_0\ d q_1$
7) $q_1 d\ R q_2$
8) $q_2 S_0\ a q_2$
9) $q_2 a\ R q_3$
10) $q_3 S_0\ b q_3$

1) **a**abc, 1) a**a**bc, 2) aa**b**c, 3) aab**c**, 4) aabc**c**, 5) aabcc,

6) aabcc**d**, 7) aabccd, 8) aabccd**a**, 9) aabccda, 10) aabccda**b**.

Пример.

P=abab, Q= ababcdab

Q=abab|cdab; S={S_0 , a, b, c, d}

1) q_0a Rq_0
2) q_0b Rq_1
3) q_1a Rq_2
4) q_2b Rq_3
5) q_3S_0 cq_4
6) q_4c Rq_5
7) q_5S_0 dq_6
8) q_6d Rq_7
9) q_7S_0 aq_8
10) q_8a Rq_9
11) q_9S_0 bq_{10}