Нормальные алгоритмы Маркова

Теория **нормальных алгоритмов** была разработана советским математиком Андреем Андреевичем Марковым в конце 1940-х годов.

При изучении разрешимости некоторых задач алгебры, он предложил новую модель вычислений, которую назвал **нормальными алгорифмами**.

Андрей Андреевич Марков (младший) (22.09.1903-11.10.1979) – советский математик, сын известного русского математика А.А.Маркова, основоположник советской школы конструктивной математики, автор понятия нормального алгоритма (1947 г.)

Нормальные алгорифмы Маркова (НАМ) — это строгая математическая форма записи алгоритмов обработки символьных строк, которую можно использовать для доказательства разрешимости или неразрешимости различных задач.

Эти алгоритмы представляют собой некоторые правила по переработке слов в каком-либо алфавите.

При этом исходные данные и результат работы алгоритма являются словами в этом алфавите.

Марков предположил, что ***любой алгоритм можно записать как НАМ***.

В отличие от машин Тьюринга **НАМ** — это "чистый” алгоритм, который не связан ни с каким "аппаратным обеспечением” (лентой, кареткой и т.п.).

НАМ преобразует одно слово (цепочку символов некоторого алфавита) в другое и задается *алфавитом* и *системой подстановок*.

Алфавитом будем называть любое непустое множество.

Его элементы называются **буквами**, а любая последовательность букв – **словами** в данном алфавите

Для удобства рассуждений допускается **пустое слово**, которые обозначим $\mathbf{\Lambda}$

Слова будем обозначать буквами P, Q, R и с индексами

Формулой подстановки называется запись вида α→β (читается «α заменить на β»), где α и β – любые слова (возможно, и пустые).

При этом **α** называется левой частью формулы, а **β** – правой частью.

Сама ***подстановка (как действие) задается формулой подстановки и*** применяется к некоторому слову *P*.

Суть операции сводится к тому, что в слове P отыскивается часть, совпадающая с левой частью этой формулы (т.е. с α), и она заменяется на правую часть формулы (т.е. на β). При этом остальные части слова P (слева и справа от α) не меняются. Получившееся слово R называют **результатом подстановки.**

Условно это можно изобразить так:

Правила выполнения НАМ

Прежде всего, задается некоторое ***входное слово P.***

Работа НАМ сводится к выполнению последовательности шагов. На каждом шаге входящие в НАМ формулы **подстановки просматриваются сверху вниз** и выбирается первая из формул, применимых к входному слову *P*, т.е. самая верхняя из тех, левая часть которых входит в *P*. Далее выполняется подстановка согласно найденной формуле. Получается новое слово *P'*.

На следующем шаге это слово *P' берется за исходное и к нему* применяется та же самая процедура, т.е. **формулы снова просматриваются сверху вниз начиная с самой верхней** и ищется первая формула, применимая к слову *P',* после чего выполняется соответствующая подстановка и получается новое слово *P''*. И так далее: $P \to P' \to P'' \to \ldots$

Следует обратить особое внимание на тот факт, что на каждом шаге формулы в НАМ всегда просматриваются начиная с самой первой.

Необходимые уточнения:

1. Если на очередном шаге была применена обычная формула $(\alpha \to \beta)$, то работа НАМ продолжается.

2. Если же на очередном шаге была применена заключительная формула $(\alpha \mapsto \beta)$, то после её применения работа НАМ прекращается. То слово, которое получилось в этот момент, и есть ***выходное слово,*** *т.е.* ***результат применения*** НАМ к входному слову.

Правила выполнения НАМ

- **Необходимые уточнения:**
  - Если левая часть α входит в P несколько раз, то на правую часть β, по определению, заменяется только первое (самое левое) вхождение α в ***P***:

P	x	α	y	α	z	$\rightarrow$	R	x	β	y	α	z

  - Если правая часть формулы подстановки – пустое слово (в таком слове нет ни одного символа), то подстановка α сводится к вычеркиванию части α из ***P*** (отметим попутно, что в формулах подстановки не принято как-либо обозначать пустое слово):

P	x	α	y	$\rightarrow$	R	x	y

  - Если в левой части формулы подстановки указано пустое слово, то подстановка $\rightarrow\beta$ сводится, по определению, к приписыванию β слева к слову *P*:

P	x	$\rightarrow$	R	β	x

Пусть дан алфавит $A = \{a_1, a_2, \dots, a_n\}$,
слово $P = a_{i_1} a_{i_2} \dots a_{i_m}$ и слово $Q = a_{j_1} a_{j_2} \dots a_{j_k}$

Под **объединением** слов PQ будем понимать слово:

$$PQ = a_{i_1} a_{i_2} \dots a_{i_m} a_{j_1} a_{j_2} \dots a_{j_k}$$

В частности: $P\Lambda = \Lambda P = P$

Кроме этого: $(P_1 P_2) P_3 = P_1 (P_2 P_3)$

Слово P является **подсловом** слова Q, если слово P является составной частью слова Q, т.е. существуют такие (возможно пустые) слова R_1 и R_2, что $Q = R_1 P R_2$.

Рассмотрим упорядоченную пару слов (P,Q)

Марковской подстановкой (P,Q) называется следующая операция над словами:
в заданном слове R находят первое вхождение слова P и, не изменяя остальных частей слова R, заменяют в нем это вхождение словом Q

Замечание:

1) Полученное слово называется **результатом** применения марковской подстановки (P,Q) к слову R

2) Если первого вхождения слова P в слово R нет (и, следовательно, вообще нет ни одного вхождения P в R), то считается что марковская подстановка (P,Q) **не применима** к слову R

Частными случаями марковских подстановок являются подстановки с пустыми словами:

(Λ, Q), (P, Λ), (Λ, Λ)

- Формула с пустой левой частью применима к любому слову. (Пустое слово всегда присутствует перед любым словом).
- Формула с пустыми левой и правой частями не меняет слово.

Для обозначения марковской подстановки (P,Q) используют запись $P \to Q$

Эту запись называют **формулой подстановки** (P,Q)

Различают **простые подстановки** $P \to Q$ и **заключительные подстановки** $P \mapsto Q$

Пример

Данное слово: **521421**

Подстановка: **21 → 3**

Результат подстановки:

5343

Пример

Данное слово: **521421**

Подстановка: **21 ı→ Λ**

Результат подстановки:

5421

Пример

Данное слово: **521421**

Подстановка: **25 → 7**

Результат подстановки:

Не применима

Пример (удаление и вставка символов)

A={*a*,*b*,*c*,*d*}. В слове *P* требуется удалить все вхождения символа *c*, а затем заменить первое вхождение подслова *bb* на *ddd*.

Например: abbcabbca → adddabba

$$\begin{cases} c \to & (1) \\ bb \to ddd & (2) \end{cases}$$

$$\text{abb}\underline{\text{c}}\text{abbca} \xrightarrow{1} \text{abbabb}\underline{\text{c}}\text{a} \xrightarrow{1} \text{a}\underline{\text{bb}}\text{abba} \xrightarrow{2} \text{addda}\underline{\text{bb}}\text{a} \xrightarrow{2}$$

$$\to \text{adddaddda}$$

$$\begin{cases} c \to & (1) \\ bb \mapsto ddd & (2) \end{cases}$$

$$\text{d}\underline{\text{c}}\text{acb} \xrightarrow{1} \text{da}\underline{\text{c}}\text{b} \xrightarrow{1} \text{dab}$$

Пример (перестановка символов)

A={*a*,*b*}. Преобразовать слово *P* так, чтобы в его начале оказались все символы *a*, а в конце – все символы *b*.

Например: babba → aabbb

Пример (использование спецзнака)

- *A*={*a*,*b*}. Удалить из непустого слова *P* его первый символ. Пустое слово не менять.

Пример (фиксация спецзнаком обрабатываемого символа)

A={0,1,2,3}. Пусть *P* – непустое слово. Трактуя его как запись неотрицательного целого числа в четверичной системе счисления, требуется получить запись этого же числа, но в двоичной системе.

Например: 0123 → 00011011

Пример (перемещение спецзнака)

A={*a*,*b*}. Требуется приписать символ *a* к концу слова *P*.

Например: bbab → bbaba

Пример (смена спецзнака)

$A=\{a,b\}$. В слове P заменить на aa последнее вхождение символа a, если такое есть. Например: babab $\to$ babaabb

Пример (перенос символа через слово)

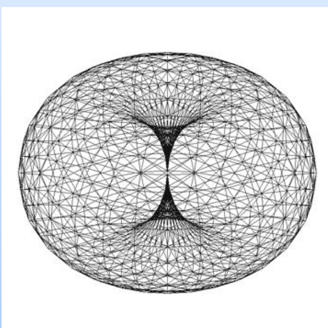
- *A*={*a*,*b*}. Перенести в конец непустого слова *P* его первый символ. Пустое слово не менять.
- Например: bbaba → babab

Пример (использование нескольких спецзнаков)

- A={a,b}. Удвоить слово P, т.е. приписать к P (слева или справа) его копию. Например: abb → abbabb

Пример (согласованная работа с различными частями слова)

Пусть слово *P* имеет чётную длину (0, 2, 4, …). Удалить правую половину этого слова. Например: bbab → bb



Создавать - лучше, чем уничтожать, а дарить - лучше, чем принимать