Алгоритмы
и
алгоритмические языки

Лекция 3

Уточнение понятия "алгоритм"

предмет → словесное описание
(последовательность символов)
Символ — любой печатный знак.
Алфавит — конечное множество символов
Слово — конечная последовательность символов из алфавита
Длина слова — количество символов в слове

Объектами алгоритма являются слова и только они.
Алгоритм — описание способа преобразования одного (входного) слова в другое (выходное)

входное слово → АЛГОРИТМ → выходное слово

Определение алгоритма: нормальные алгоритмы Маркова

1950-е гг. советский математик А.А.Марков предложил еще одно формальное определение алгоритма, названное нормальные алгоритмы Маркова

А.А. Марков (1903-1979)

Подстановки

Формула подстановки – это запись вида $\alpha \to \beta$, где α и β – слова из нек. алфавита (возможно пустые)

Подстановка – действие, задается формулой подстановки и применяется к слову P.

1. Если левая часть ФП входит в слово P, то ФП **применима** к P, если левая часть ФП не входит в P, то ФП **неприменима** к P.
2. Если левая часть ФП входит в слово P несколько раз, то заменяется одно первое (левое) вхождение.
3. $\alpha \to$ – вычеркивание одного α из P
4. $\to \beta$ – приписывание β слева к слову (эта ФП применима к любому слову)

Нормальный алгоритм Маркова

Нормальным алгоритмом Маркова (НАМ) называется конечный упорядоченный набор формул подстановки

$\alpha_1 \longrightarrow \beta_1$

$\alpha_2 \longrightarrow \beta_2$

…

$\alpha_n \longrightarrow \beta_n$

Выполнение алгоритма

1. Задаем входное слово
2. Просматриваем формулы подстановки сверху вниз, выбираем первую применимую, выполняем подстановку,
3. К результату подстановки применяем шаг 2.

В каких случаях процесс применения правил завершится?

Примеры НАМ

Пример 1. A={a,b,c}, требуется вместо P получить такое количество |, сколько букв c входит в P.

Пример 2. A={a,b,*}, P = *Q, Q – слово из a,b. Получить Q* (переставить * в конец слова).

Пример 3. A={a,b}. Добавить в конец P символ a. (Например, из ***abb*** получить ***abba***).

Пример 4. A={a,b,c}, требуется удвоить последнее вхождение a в P.
(Если a в P не входит, не менять P).

Принцип нормализации
(Тезис Маркова)

Любой алгоритм нормализуем,

т. е. для любого алгоритма можно написать эквивалентный ему нормальный алгоритм Маркова

Алгоритмически неразрешимые проблемы

Задача называется алгоритмически разрешимой, если существует алгоритм ее решения.

Задача называется алгоритмически неразрешимой, если не существует алгоритма ее решения.

На примере НАМ рассмотрим три алгоритмически неразрешимые проблемы:

1. Проблема самоприменимости
2. Проблема останова
3. Проблема эквивалентности

Проблема самоприменимости

НАМ:

Запись алгоритма:

$\alpha 1 \rightarrow \beta 1;\ \alpha 2 \rightarrow \beta 2;\ \ldots;\ \alpha n \rightarrow \beta n$

$\boldsymbol{\alpha_1 \longrightarrow \beta_1}$

$\boldsymbol{\alpha_2 \longrightarrow \beta_2}$

…

$\boldsymbol{\alpha_n \longrightarrow \beta_n}$

Алгоритмы, которые применимы к своей записи, называются самоприменимыми.

Проблема самоприменимости

НАМ:

$\alpha_1 \longrightarrow \beta_1$

$\alpha_2 \longrightarrow \beta_2$

…

$\alpha_n \longrightarrow \beta_n$

Запись алгоритма:

$\alpha 1 \rightarrow \beta 1;\ \alpha 2 \rightarrow \beta 2;\ \ldots;\ \alpha n \rightarrow \beta n$

Алгоритмы, которые применимы к своей записи, называются самоприменимыми.

Теорема. Не существует алгоритма А, который для любого алгоритма Е по его записи мог бы определить, является ли Е самоприменимым или нет.

Проблема останова

Теорема.

Не существует алгоритма, который бы по записи произвольного алгоритма E и слову P определял бы, является ли E применимым к P.

Проблема эквивалентности

Алгоритмы B и C эквивалентны в алфавите A (на множестве A*), если к каждому слову из A* они
– либо одновременно неприменимы,
– либо одновременно применимы и выдают одинаковое выходное слово.

Теорема.
Не существует алгоритма, который для заданных двух алгоритмов по их записи определяет, являются они эквивалентными (в заданном алфавите) или нет.
(рассматриваем без доказательства)