Алгоритмы и алгоритмические языки

конспект от TheLostDesu

7 сентября 2021 г.

1 Алгоритм

1.1 Определение

Под алгоритмом(или эффективной процедурой) в математике понимают точное предписание, задающее вычислительный процесс, ведущий от начальных данных, которые могут варироватся, к искомому результату.
Алгоритм должен обладать следующими свойствами:
Конечность. Алгоритм должен заканчиватся за конечное число шагов
Определенность[1]. Каждый шаг и переход от шага к шагу должны быть точно определены, и каждое применение алгоритма к одним и тем же исходным данным должно приводить к одинаковому результату.[2]
Простота и понятность. Каждый шаг должен быть четко и ясно определен, желательно самыми простыми командами, чтобы алгоритм мог выполнить любой исполнитель.
Массовость. Алгоритм должен представлять общий метод решения класса задач.

1.2 Пример

Алгоритм Евклида нахождения НОД двух чисел
Даны 2 числа, a и b. Цель - найти НОД.
Решение:
Выполнить следующие шаги:
1. Если $a < b$: поменять их местами
2. Разделить нацело a на b; Получить остаток r.

[1] Также называют детерминированностью

[2] Иногда, например при работе с адресами в памяти невозможно добиться детерминированности

3. Если r = 0, то НОД(a, b) = b
4. Иначе заменить a на b; b на r; Вернуться к шагу 2.

1.3 Формализация понятия алгоритма

Не имея формального определения, невозможно доказать, что задача неразрешима алгоритмически.
Существует тезис Тьюринга-Чёрча, гласящий что любая интуитивно вычисляемая функция имеет алгоритмическое решение. Однако он формально недоказан.

Алфавит - конечное множество A_p элементов a_j: $A_p = a_1, a_2 ... a_p$[3]
Элементы алфавита называются символами.
Последовательность из m символов алфавита A_p называется словом длины m над алфавитом A_p Длина слова обозначается как |m|
Слово длины 0 называется пустым словом и обозначается ε
Множество всех слов над алфавитом A_p обозначается как A_p^*[4]

Утверждение. Для любой пары алфавитов A и B можно выполнить кодирование алфавита A с помощью алфавита B и обратно, возможно, с применением дополнительного символа "конца слова".

Следствие. При доказательстве чего-либо можно ограничится одним алфавитом(Например A_2[5])

Алгоритм, тогда, по факту является частичным отображением одного слова из множества A^* в другое слово из множества A^*

Утверждение. Существует взаимно-однозначное отображние $A^* \leftrightarrow N_0$, то есть возможно пронумеровать все слова. Тогда станет возможно по номеру получить слово(операция обозначается за решеточку), и по слову получить номер(операция обозначается решеточка в минус первой степени)
Это значит, что алгоритму можно поставить в соответствие функцию, которая переводит натуральные числа в натуральные числа.
Тогда, каждая частично определенная функция переводящяя натуральные числа в натуральные числа определяет какой-либо алгоритм.

[3]Алфавит может состоять из чего угодно: Букв, цифр, символов, объектов(?)

[4]По сути, является объединением всех возможных слов алфавита. Слова не бывают бесконечной длины, но длина может быть любой конечной величиной.

[5]Алфавитом из двух символов

2 Машина-автомат

Назовем машину, которая получая любое исходное слово $w \in A^*$ выдает другое слово v - машиной-автоматом.

Машина тьюринга

Пусть у нас есть лента бесконечной длины[6]. Пустые клетки в ней обозначаются за λ. Машина может оперировать алфавитом $S = AandA'$.
A - алфавит входных символов
A' - алфавит вспомогательных символов(маркеров).
Пусть есть "Управляющая головка головка, которая может писать что-то в "Рабочую ячейку". Она также может считывать то, что записано на ленте и выполнять действия, записаные где-то.
Но этого может не хватить, ведь машина тогда будет ограниченно размерами алфавита. поэтому решение того, что делает головка вводится также "Алфавит состояний множество, описывающее состояние машины. Состояние традиционно обозначается за q. Тогда за такт машина сможет:
1) Сменить состояние
2) Записать в ячейку
3) Переместить головку на n символов
4) Закончить выполнение программы[7]. В любой такт машине известна пара значений: то, что записанно на ленте, состояние из алфавита.

2.1 Примеры

Проверка скобочной последовательности на правильность.
Правильная скобочная последовательность[8]
1) число скобок открывающих, равно числу закрывающих
2) каждая закрывающая скобка соответствует более ранней открывающей
Дается какая-то скобочная последовательность. Записать 0 на ленту, если она правильная, 1 если нет. Решение: 1. Идти направо, до ближайшей закрыващей скобки
2. Идти налево до открывающей скобки
3. Пометить открывающюю скобку(Записать X в ячейку вместо нее)
4. Вернутся к закрывающей скобке
5. Повторять, пока есть и закрывающие и открывающие скобки

[6]Причина, по которой машина тьюринга мощнее современных компьютеров - бесконечность ленты.
[7]Достигается в завершающем состоянии
[8]Для одного типа скобок

6. Если скобок не осталось - записать 1
7. Иначе записать 0

Обычно машину можно описать в виде таблицы. Строки - номер состояния. Столбцы - то, что написанно на ленте. В ячейках записано: то, в какое состояние перейти, то, что написать на ленту, влево или вправо ли переместится головке.