Лекция 1

Эффективная вычислимость

Рекурсивные функции

Алгоритм – некоторое формальное предписание, действуя согласно которому можно получить решение задачи. Все определения алгоритмов (Марков, Черч, Тьюринг) имеют общие свойства.

Свойства алгоритма

1. **Элементарность** шагов алгоритма: решение задачи разбивается на этапы.
2. **Определенность:** на каждом шаге известно, что делать на следующем шаге.
3. **Направленность:** имеется цель алгоритма.
4. **Массовость:** алгоритм служит для решения целого ряда однотипных задач.

Определение 1. Функция $f(x_1, x_2, \dots, x_n)$ – эффективно вычислима, если существует алгоритм, согласно которому каждому набору значений переменных $< c_1, c_2, \dots, c_n >$ из множества M ставится в соответствие значение функции

$f(c_1, c_2, \dots, c_n) = c, c_i \in M, c \in M.$

Примитивно-рекурсивные функции

В этом разделе будем рассматривать функции, определенные на множестве

$\tilde{\mathbb{N}} = \mathbb{N} \cup \{0\}$ или его подмножествах.

Функция всюду определена, если область определения переменных – $\tilde{\mathbb{N}}$. Функция частично определена, если область определения переменных – подмножество $\tilde{\mathbb{N}}$.

Простейшие функции:

1. *Обнуление:* $O(x) = 0$.

2. *Сдвиг* (прибавление 1): $S(x) = x + 1$.

3. *Функция выбора, или проектирование*: $I_m(x_1, \dots, x_n) = x_m$.

Эти функции всюду определены.

Операторы:

1. ***Суперпозиции*** – подстановка вместо какой-нибудь переменной функции другой функции (см. прошлый семестр: $f_1(x_1, \dots, f_2(x_1, \dots, x_{n_2}), \dots, x_{n_1})$.

Точнее, говорят, что n-местная функция $\psi(x_1,\dots,x_n)$ получена с помощью оператора суперпозиции из m-местной функции $\varphi(x_1,\dots,x_m)$ и n-местных функций $f_1(x_1,\dots,x_n),\dots,f_m(x_1,\dots,x_n)$, если

$$\psi(x_1,\dots,x_n)=\varphi\big(f_1(x_1,\dots,x_n),\dots,f_m(x_1,\dots,x_n)\big).$$

2. ***Примитивной рекурсии*** – некоторая схема, по которой можно получить следующее значение функции, используя предыдущее. Опишем подробно этот оператор. Рекурсию будем проводить по переменной y.

   Функция $f(x_1,\dots,x_n,y)$ содержит $(n+1)$ переменную; $\varphi(x_1,\dots,x_n)-n$ переменных; $\psi(x_1,\dots,x_n,\mathrm{y},\mathrm{z})-(n+2)$ – переменных.

   Схема примитивной рекурсии:

   $$\begin{cases} f(x_1,\dots,x_n,0)=\varphi(x_1,\dots,x_n) \\ f(x_1,\dots,x_n,y+1)=\psi(x_1,\dots,x_n,\mathrm{y},\mathrm{z}) \end{cases}, \quad \text{где } z=f(x_1,\dots,x_n,y).$$

   Если функции $\varphi(x_1,\dots,x_n)$ и $\psi(x_1,\dots,x_n,\mathrm{y},\mathrm{z})$ примитивно рекурсивны, то и $f(x_1,\dots,x_n,y)$ примитивно рекурсивна.

Выделим частный случай: $n=0$

$$\begin{cases} f(0)=const \\ f(y+1)=\psi(\mathrm{y},\mathrm{z}) \end{cases}.$$

Определение 2. Класс функций, которые можно получить из простейших с помощью конечного числа применений операторов суперпозиции и примитивной рекурсии называется классом примитивно-рекурсивных функций. Функции этого класса называются примитивно-рекурсивными.

Примеры 1-9. Примитивно-рекурсивные функции.

Используем только оператор суперпозиции.

1. $x+n=\underbrace{S(\dots S(x)}_{n}\dots)\ \ \forall n\in\widetilde{\mathbb{N}},n\geq 1$
2. $n=\underbrace{S(S(O(x))}_{n}\dots)$

   Оператор примитивной рекурсии.

3. $\sigma(x,y)=x+y$

   $$\begin{cases} f(x,0)=x=I_1(x) \\ f(x,y+1)=x+(y+1)=(x+y)+1=z+1=S(z)=S(I_3(x,y,z))' \end{cases}$$

   $z=f(x,y).$

4. $P(x,y) = x \cdot y$

$$\begin{cases} P(x,0) = x \cdot 0 = 0 = O(x) \\ P(x,y+1) = x(y+1) = \underbrace{xy}_{z} + x = z + x = \sigma(x,z) = \sigma(I_1(x,y,z), I_3(x,y,z)) \end{cases}$$

5. **Пример РГР №1**.

Доказать, что заданная функция примитивно-рекурсивна, используя примитивную рекурсивность суммы $\sigma(x,y) = x + y$. $\boldsymbol{f(x,y) = 2(x+1)(y+1)}$

$$\begin{cases} f(x,0) = 2(x+1) = (x+1) + (x+1) = S(x) + S(x) = \sigma\big(S(x), S(x)\big) \\ f(x,y+1) = 2(x+1)(y+1+1) = 2\underbrace{(x+1)(y+1)}_{z} + 2(x+1) = \\ = z + \sigma\big(S(x), S(x)\big) = \sigma\Big(z, \sigma\big(S(x), S(x)\big)\Big) = \sigma\Big(I_3(x,y,z), \sigma\big(S(I_1(x,y,z)), S(I_1(x,y,z))\big)\Big) \end{cases}$$

6. $f(x,y) = x^y$

$$\begin{cases} f(x,0) = x^0 = 1 \\ f(x,y+1) = x^{y+1} = x^y \cdot x = P(x,z) \end{cases}$$

7. $x \dot{-} 1 = \begin{cases} x-1, & x \geq 1 \\ 0, & x = 0 \end{cases}$

$$\begin{cases} 0 \dot{-} 1 = 0 \\ (x+1) \dot{-} 1 = x + 1 - 1 = x = I_1(x) \end{cases}$$

8. $x \dot{-} y = \begin{cases} x-y, & x \geq y \\ 0, & x < y \end{cases}$

$$\begin{cases} x \dot{-} 0 = x = I_1(x) \\ x \dot{-} (y+1) = (x \dot{-} y) \dot{-} 1 = z \dot{-} 1, \end{cases} \qquad z = x \dot{-} y;$$

Легко убедиться, что равенство выполняется для любых x и y:

$x > y \Rightarrow x - y - 1 = x - y - 1$

$x \leq y \Rightarrow 0 = 0$

9. $|x - y| = (x \dot{-} y) + (y \dot{-} x)$

Задача 1. Доказать самостоятельно, что функции $Max(x,y)$ и $Min(x,y)$ – примитивно-рекурсивны.

Пример 10. Задана примитивно рекурсивная функция $f(x,y)$ схемой примитивной рекурсии.

1. Найти $f(4,3)$, если

$$\begin{cases} f(4,0) = 2 \\ f(4,y+1) = zx \end{cases}$$

Решение. Воспользовавшись схемой примитивной рекурсии, получим

$$f(4,1) = f(4,0) \cdot 4 = 2 \cdot 4 = 8$$

$$f(4,2) = f(4,1) \cdot 4 = 8 \cdot 4 = 32$$

$$f(4,3) = f(4,2) \cdot 4 = 32 \cdot 4 = 128$$

2. Найти $f(x,y)$, если

$$\begin{cases} f(x,0) = x \\ f(x,y+1) = zx^2 \end{cases}$$

Решение. Распишем по шагам согласно схеме примитивной рекурсии, получим

$$\begin{cases} f(x,0) = x \\ f(x,1) = zx^2 = f(x,0)x^2 = xx^2 = x^3 \\ f(x,2) = zx^2 = f(x,1)x^2 = x^3x^2 = x^5 \\ \dots \\ f(x,y) = zx^2 = f(x,\mathrm{y}-1)x^2 = x^{2y-2+1}x^2 = x^{2y+1} \end{cases}$$

Частично рекурсивные функции

Частично рекурсивные функции выводят из класса всюду определенных функций. Частично рекурсивные функции определены на некотором подмножестве $M \subseteq \widetilde{\mathbb{N}}$.

Примеры частично-рекурсивных функций:

1. $x - y = \begin{cases} x - y, & x \geq y \\ \text{не определено}, & x < y \end{cases}$

2. $\frac{x}{y} = \begin{cases} \frac{x}{y}, & x \vdots y \\ \text{не определено в остальных случаях} \end{cases}$

Простейшие функции:

1. Обнуление: $O(x) = 0$.

2. Сдвиг (прибавление 1): $S(x) = x + 1$.

3. Функция выбора, или проектирование: $I_m(x_1, \dots, x_n) = x_m$.

Операторы:

1. *Суперпозиция*
2. *Примитивная рекурсия*
3. ***Минимизация.***

Определение 3. Говорят, что функция $f(x_1,\dots,x_n)$ получена с помощью опрератора минимизации

$$f(x_1,\dots,x_n)=\mu_y[g(x_1,\dots,x_n,y)=0]\text{ , если}$$

$f(x_1,\dots,x_n)$ определена и $f(x_1,\dots,x_n)=y \Leftrightarrow$ определены и $\begin{matrix} g(x_1,\dots x_n,0)\neq 0 \\ g(x_1,\dots x_n,1)\neq 0 \\ g(x_1,\dots x_n,y-1)\neq 0 \\ g(x_1,\dots x_n,y)=0 \end{matrix}$

Пример 11. Задана функция $g(x_1,x_2,y)$, найти $f(4,3)$, если

$$g(4,3,0)\neq 0$$

$$g(4,3,1)\neq 0$$

$$g(4,3,2)=0\Rightarrow f(4,3)=2$$

Пример 12.

1. Используя оператор минимизации доказать, что функция частично рекурсивна.

$$x-y=\begin{cases} x-y, & x\geq y \\ \text{не определена при } x<y \end{cases}$$

Построим функцию $g(\mathrm{x},y,z)$ следующим образом

$$x-y=z\Rightarrow x-y-z=0$$

$$g(\mathrm{x},y,z)=\underbrace{|x-(y+z)|}_{g(x,y,z)}$$

Тогда $x-y=\mu z[|x-(y+z)|=0]$.

Проверка

$x<y \qquad |x-(y+z)|\neq 0$

$z=x-y\Rightarrow |x-(y+z)|=0$

2. Используя оператор минимизации доказать, что функция частично рекурсивна.

$$\frac{x}{y} = \begin{cases} \frac{x}{y}, x \vdots y \\ \text{не определено в иных случаях} \end{cases}$$

Построим функцию $g(\mathrm{x}, y, z)$

$$\frac{x}{y} = z \Rightarrow x - yz = 0$$

$$g(\mathrm{x}, y, z) = \underbrace{|x - yz|}_{g(x,y,z)}$$

Тогда $\frac{x}{y} = \mu z[|x - (yz)| = 0]$.

Определение 4. Класс частично рекурсивных функций — это класс функций, полученных из простейших за конечное число применений операторов суперпозиции, примитивной рекурсии и минимизации. Функции этого класса называются частично рекурсивными.

Тезис Чёрча: Класс частично-рекурсивных функций совпадает с классом эффективно вычислимых функций.

Эффективно вычислимые функции можно реализовать с помощью машины Тьюринга.

Сложность алгоритмов, содержащих рекурсию

Алгоритмы, содержащие рекурсию, могут иметь различную сложность. Так алгоритм вычисление примитивно-рекурсивной функции $n!$ имеет сложность $O(n)$, и соответственно программа содержит один цикл.

Доказано в примере 6, что функция x^n (x^y) является примитивно-рекурсивной. Если для нахождения ее значений использовать рекурсию (вызывать подпрограмму умножения чисел $(n-1)$ раз) получим: вычислительная сложность $O(n-1)$. Можно уменьшить сложность алгоритма до $O(log_2 n)$, получая значения $x^2, (x^2)^2, \ldots$ с соответствующей коррекцией на близкие степени.

Этот небольшой пример показывает: не используя рекурсию, можно уменьшить вычислительную сложность алгоритма. Во многих книгах по программированию рекомендуется по возможности рекурсию избегать. Заметим, что нахождение вычислительной сложности алгоритма, содержащего рекурсию, часто задача очень трудная.

Одним из способов избежать рекурсивную процедуру и тем самым значительно уменьшить вычислительную сложность алгоритма является доказательство простой формулы, позволяющей получить тот же результат за более короткое время.

В качестве примера возьмем вычисление чисел Фибоначчи. Числа Фибоначчи — бесконечная числовая последовательность Φ_n, в которой каждое число есть сумма двух предыдущих: $\Phi_n = \Phi_{n-1} + \Phi_{n-2}$. Первые два элемента последовательности, нужные для затравки — ноль и единица: $\Phi_0 = 0, \Phi_1 = 1$. Вот числа Фибоначчи из первой сотни: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, …

Доказано, что числа Фибоначчи получаются по *формуле Бине:*

$$\Phi_n = \frac{\varphi^n - \psi^n}{\sqrt{5}},$$

где $\varphi = \frac{1+\sqrt{5}}{2}$ =1,61803398874989…— *число Фидия*, $\psi = \frac{1-\sqrt{5}}{2} = -\frac{1}{\varphi}$. Удивительно здесь то, что, несмотря на присутствие иррационального числа Фидия, формула даёт целые числа. На больших значениях чисел Фибоначчи может появиться ошибка от округления $\sqrt{5}$. Но, как мы уже видели, сложность процедуры возведения числа в степень n очень небольшая.

Если же для нахождения чисел Фибоначчи использовать непосредственно формулу, т.е. воспользоваться рекурсивным алгоритмом, то нам придется для вычисления одного числа последовательности вычислять и предыдущие значения (рис 1.)

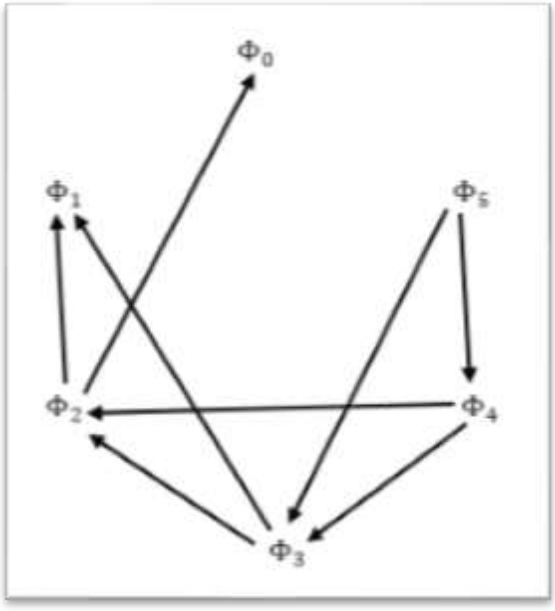

Рис. 1. Числа Фибоначчи

Глядя на рисунок, можно подсчитать, сколько раз вызывалась рекурсивная процедура для вычисления Φ_5: 14 раз. Доказано, что в общем случае число обращений составляет геометрическую прогрессию со знаменателем φ. Таким образом, алгоритм будет иметь экспоненциальную сложность (принадлежать к классу NP).

http://mech.math.msu.su/~shvetz/54/inf/perl-problems/chFibonacci.xhtml