

Алгоритмы и структуры данных, лекция 1

12.01.2016

Оргмоменты

Одна контрольная — констест на реализацию какого-то алгоритма. Пользоваться своим кодом запрещено.

Задача — привести алгоритм, провести теоретический анализ (доказать его корректность, оценить время работы) и запрограммировать. Сначала сдаётся теория, потом практика.

Экзамен устный.

$$O_{\text{итоговая}} = 0.7 \cdot O_{\text{накопленная}} + 0.3 \cdot O_{\text{экзамен}}$$
$$O_{\text{накопленная}} = 0.2 \cdot O_{\text{КР}} + 0.12 \sum_{i=1}^5 O_{\text{ДЗ } i} + 0.2 \cdot O_{\text{семинары}}$$

Списывание, как обычно, *не поощряется*. ДЗ предполагается не обсуждать.

[Здесь](#) (ссылка слева) можно найти ссылки на ДЗ и краткое содержание лекций.

Автоматов *пока* не предусмотрено.

Литература:

- Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. — «Алгоритмы. Построение и анализ»
- Дасгупта С., Пападимитриу Х., Вазирани У. — «Алгоритмы»

Лекция

Ханойские башни

Есть три стержня. На первый стержень нанизано 64 диска, от самого большого к самому маленькому. Задача: переложить все диски на второй стержень. Ограничения:

- Диски можно переносить только по одному.
- Нельзя класть диск большего диаметра на диск меньшего диаметра.

Какой может быть алгоритм?

Варианты из аудитории:

1. Полный перебор
2. Рекурсивный алгоритм.

Рассмотрим такой рекурсивный алгоритм:

1. Переложим все диски, кроме n -ного, на третий стержень;
2. Переложим n -ный диск с первого на второй стержень;

3. Переложим все остальные с третьего стержня на второй.

Запишем этот алгоритм с помощью псевдокода:

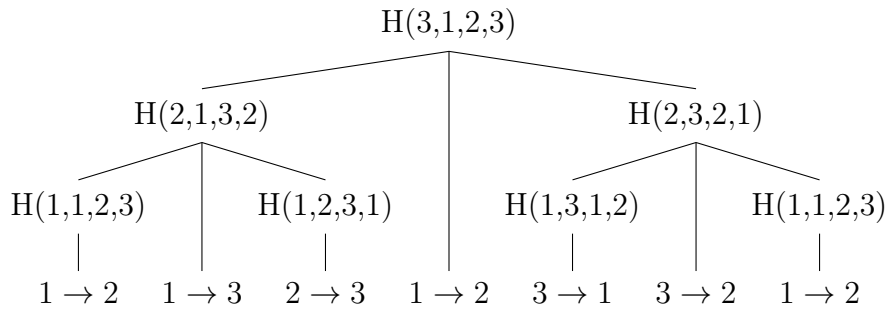
Algorithm 1 Рекурсивный алгоритм решения задачи о Ханойской башне

```

1: function HANOI3( $n, i, j, k$ ) ▷  $n$  — количество дисков,  $i, j, k$  — номера стержней
2:   if  $n > 0$  then
3:     HANOI3( $n - 1, i, k, j$ )
4:     move  $i \rightarrow j$ 
5:     HANOI3( $n - 1, k, j, i$ )

```

Нарисуем дерево операций для $n = 3$:



Алгоритм, по сути, обходит это дерево в глубину и при этом слева направо, выполняя все перемещения, что встретятся.

Это дерево можно рассматривать, как полное бинарное дерево глубины n , если перемещения учитывать не в отдельных листьях, а в родительских узлах. Тогда в каждом узле мы выполняем одно действие, а в полном бинарном дереве $2^n - 1$ узлов. Следовательно, выполняется $2^n - 1$ перемещение.

Пусть число перемещений для n дисков равно $f(n)$. Тогда верно следующее:

$$f(n) = \begin{cases} 0, & n = 0 \\ 2f(n-1) + 1, & n > 0 \end{cases}$$

Свойство: $f(n) = 2^n - 1$

Доказательство. Докажем это по индукции. База верна, так как $f(0) = 0 = 2^0 - 1$. Теперь пусть предположение верно для $n - 1$, то есть $f(n - 1) = 2^{n-1} - 1$. Тогда $f(n) = 2f(n - 1) + 1 = 2(2^{n-1} - 1) + 1 = 2^n - 2 + 1 = 2^n - 1$, что и требовалось доказать. \square

Можно ли улучшить время работы? Оказывается, что нет.

Рассмотрим некоторый алгоритм. Он рано или поздно должен переложить наибольший диск на второй стержень. Для этого ничего не должно быть на нём и на втором, т.е. все на третьем. А как получить эту конфигурацию? Оптимальным алгоритмом на $n - 1$ шаг, что приводит к нашим вычислениям и уже полученному минимальному результату в $2^n - 1$.

Утверждение. Задачу о Ханойских башнях нельзя решить за меньшее число шагов, причём решение с таким числом шагов ровно одно.

Доказательство. По индукции.

База ($n = 0$): очевидно, решить быстрее, чем за 0 шагов нельзя и последовательность такая ровно одна.

Переход ($n - 1 \rightarrow n$): предположим, что мы доказали это утверждение для $n - 1$. Рассмотрим утверждение для n . Рано или поздно алгоритму понадобится освободить первые два стержня, чтобы переложить первый диск на второй стержень. Необходимо сделать это одно перекалывание и после вернуть все оптимальным алгоритмом. Итого, опираясь на предположение индукции шагов в оптимальном и единственном решении для $n - 1$, нам понадобится $2(2^{n-1} - 1) + 1$ шаг, что и равно $2^n - 1$. \square

Изменим задачу:

Четыре стержня

Условие в остальном ровно то же. Стала ли задача проще?

Сложнее она точно не стала, т.к. четвёртым можно просто не пользоваться.

Рассмотрев переход от двух к трём, кажется, что должно быть проще; как можно воспользоваться четвёртым?

Предложение: переложить предпоследний отдельно на четвёртый и сэкономить на перекалывании башни из $n - 1$, перекалывая вместо неё башню из $n - 2$

Algorithm 2 Рекурсивный алгоритм решения задачи о Ханойской башне на 4-х стержнях, версия 1

```

1: function HANOI4( $n, i, j, k, l$ )            $\triangleright n$  — количество дисков,  $i, j, k, l$  — номера стержней
2:   if  $n > 0$  then
3:     HANOI4( $n - 1, i, l, k, j$ )
4:     move  $i \rightarrow k$ 
5:     move  $i \rightarrow j$ 
6:     move  $k \rightarrow j$ 
7:     HANOI4( $n - 1, l, j, i, k$ )

```

Построив аналогичное дерево, получим, что в каждом узле три перемещения, а узлов $2^{\lfloor \frac{n}{2} \rfloor} - 1$ — экономия, но не очень большая.

Построим другой алгоритм:

Algorithm 3 Рекурсивный алгоритм решения задачи о Ханойской башне на 4-х стержнях, версия 2

```

1: function HANOI4( $n, i, j, k, l$ )            $\triangleright n$  — количество дисков,  $i, j, k, l$  — номера стержней
2:   if  $n > 0$  then
3:     HANOI4( $n - m, i, l, k, j$ )
4:     HANOI3( $m, i, j, k$ )
5:     HANOI4( $n - m, l, j, i, k$ )

```

Заметим, что число шагов зависит от m . Пусть $n_m = \frac{m(m+1)}{2}$ (если n другое, то на первом шаге выберем такой m , чтобы $n - m$ было таким, а дальше на вход будет поступать число такого вида).

Построим дерево алгоритма. Оно также будет полным бинарным деревом. Заметим, что в нём m уровней, так как при каждом шаге m уменьшается на единицу. Это связано с тем, что $n_m - m = \frac{m(m+1)}{2} - m = \frac{(m-1)m}{2} = n_{m-1}$. При этом в узле на i -м уровне (нумерация с нуля) проводится $2^{m-i} - 1$ операция, так как мы пользуемся доказанным ранее алгоритмом для трёх стержней. Тогда на каждом уровне выполняется $2^m - 2^i$ операций. Тогда всего выполняется $\sum_{i=0}^{m-1} (2^m - 2^i) = m2^m - \sum_{i=0}^{m-1} 2^i = (m-1)2^m + 1$ операций.

Пусть число перемещений для n дисков равно $g(n)$. Тогда

$$g(n_m) = \begin{cases} 0, & m = 0 \\ 2g(n_{m-1}) + 2^m - 1, & m > 0 \end{cases}$$

Предложение: $g(n_m) = (m - 1)2^m + 1$

Доказательство. По индукции. База верна, так как $g(n_0) = 0 = (0 - 1)2^0 + 1 = -1 + 1$. Теперь допустим, что предположение верно для n_{m-1} , то есть $g(n_{m-1}) = (m - 2)2^{m-1} + 1$. Тогда $g(n_m) = 2g(n_{m-1}) + 2^m - 1 = (m - 2)2^m + 2 + 2^m - 1 = (m - 1)2^m + 1$. \square

Заметим, что при достаточно больших n верно, что $m \approx \sqrt{2n}$. Тогда $g(n) \approx \sqrt{2n} \cdot 2^{\sqrt{2n}}$. Тогда $g(n) = \Theta(\sqrt{n} \cdot 2^{\sqrt{2n}})$. Θ означает (грубо говоря), что функция растёт примерно так же.

Попробуем обобщить этот алгоритм для любого числа стержней:

Algorithm 4 Рекурсивный алгоритм решения задачи о Ханойской башне, общий случай

```

1: function HANOI( $n, i, j, P$ )                                 $\triangleright n$  — количество дисков,  $i, j$ , — основные стержни
2:                                                          $\triangleright P$  — множество вспомогательных стержней
3:   if  $n > 0$  then
4:     choose  $p \in P$ 
5:      $R := P \setminus p$ 
6:     if  $R = \emptyset$  then
7:       HANOI3( $n, i, j, p$ )
8:     else
9:       HANOI( $n - m, i, p, R \cup \{j\}$ )
10:      HANOI( $m, i, j, R$ )
11:      HANOI( $n - m, p, j, R \cup \{i\}$ )

```

Пусть для перемещения n дисков с помощью алгоритма на k стержнях ($k \geq 3$) нужно $h(n, k)$ операций. Тогда верно следующее:

$$h(n_m, k) = \begin{cases} 0 & n = 0 \\ 2^{n_m} - 1 & n > 0, k = 3 \\ 2h(n_{m-1}, k) + h(m, k - 1) & n > 0, k > 3 \end{cases}$$