Алгоритмы и структуры данных, лекция 1

12.01.2016

Оргмоменты

Одна контрольная — контест 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;

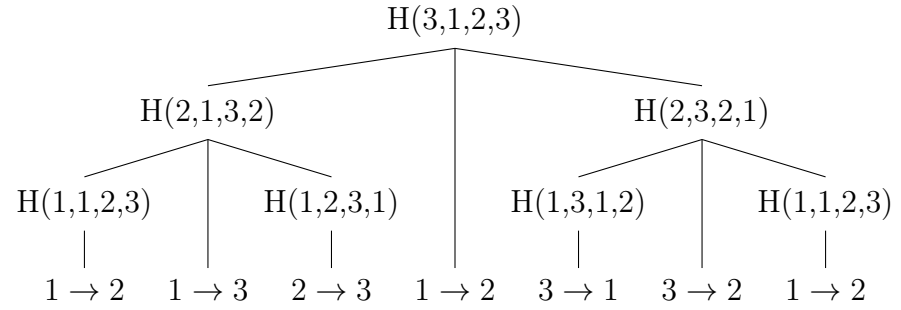
3. Переложим все остальные с третьего стержня на второй.

Запишем этот алгоритм с помощью псевдокода:

Algorithm 1 Рекурсивный алгоритм решения задачи о Ханойской башне

```
1: function HANOI3(n, i, j, k)
2:     if n > 0 then
3:         HANOI3(n − 1, i, k, j)
4:         move i → j
5:         HANOI3(n − 1, k, j, i)
```

Нарисуем дерево операций для $n = 3$:



Алогритм, 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. □

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 $(n-1 \to n)$**:** 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 перекладывание 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. □

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?

Предложения:

- Переложить предпоследний отдельно на четвёртый и сэкономить на перекладывании башни из $n-1$, перекладывая вместо неё башню из $n-2$

```
Hanoi4(n,i,j,k,l)
    if n>0 then
        Hanoi4(n-2, i, k, j, l)
        move i to l
        move i to j
        move l to j
        Hanoi4(n-2, k, j, i, l)
```

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

Построим другой алгоритм:

```
Hanoi4(n,i,j,k,l)
    if n>0 then
        Hanoi4(n-m, i, l, j, k)
        Hanoi3(n, i, j, k)
        Hanoi4(n-m, l, j, i, k)
```

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

Получим то же двоичное дерево, но заметим, что число уровней в нём — m, т.к. на каждом шаге m уменьшается на 1 (ПОЧЕМУ? пояснить!), а в каждом узле работы проводится 2^m-1, т.к. мы пользуемся уже доказанным алгоритмом для трёх стержней.

Сложим по уровням. На i-ом уровне сумма всех узлов — 2^m-2^i, при нумерации с нуля.

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

$$g(n) = \sum_{i=0}^{m-1}(2^m - 2^i) = m*2^m - \sum_{i=0}^{m-1} 2^i = m*2^m - (2^m-1) = (m-1)\cdot 2^m + 1$$

Докажем по индукции:

База: $m=0; g(n_0) = 0 = -1\cdot 2^0 + 1$

Переход: $g(m-1) = (m-2)\cdot 2^{m-1}+1$ $g(m) = 2((m-2)\cdot 2^{m-1}+1)$!!!!! СПИСАТЬ

$m \approx \sqrt{2n}; g(n) \approx \sqrt{2n}\cdot 2^{\sqrt{2n}}$

$g(n) = \Theta(\sqrt{2n}\cdot 2^{\sqrt{2n}})$

Попробуем обобщить этот алгоритм для любого числа стержней:

```
Hanoi(n, i,j,P)
    if n>0 then
        choose p \in P
        R := P\\{p\}
        if R\neq\varnothong then
            Hanoi3(n, i, j, p)
        else
            Hanoi(n–m, i, p, R\cup\{j\})
            Hanoi(m, i, j, k)
            Hanoi(n–m, )<++>
```

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