Лабораторная работа № 8 по курсу дискретного анализа: Жадные алгоритмы

Выполнил студент группы М8О-307Б-20 *Белоусов Егор Владимирович*

Условие

Разработать жадный алгоритм решения задачи, определяемой своим вариантом. Доказать его корректность, оценить скорость и объём затрачиваемой оперативной памяти.

Реализовать программу на языке C или C++, соответствующую построенному алгоритму. Формат входных и выходных данных описан в варианте задания.

На первой строке заданы два числа, N и $p > 1$, определяющие набор монет некоторой страны с номиналами $p^0, p^1, ..., p^{N-1}$. Нужно определить наименьшее количество монет, которое можно использовать для того, чтобы разменять заданную на второй строчке сумму денег $M \leq 2^{32} - 1$ и распечатать для каждого i-го номинала на i-ой строчке количество участвующих в размене монет. Кроме того, нужно обосновать почему жадный выбор неприменим в общем случае (когда номиналы могут быть любыми) и предложить алгоритм, работающий при любых входных данных.

Метод решения

Представим задачу в более математическом виде: требуется выразить число M в виде $M = \sum_{i=0}^{N-1} \alpha_i \cdot p^i$, где $\alpha_i \in N \cup 0$ так, чтобы $\sum_{i=0}^{N-1} \alpha_i$ была минимальна. Решение базируется на следующем утверждении.

Утверждение. Для любого α_i при $i \neq N - 1$, выполняется $\alpha_i < p$.
Это верно, так как если $\alpha_i \geq p$, то всегда возможно заменить α_i на $\alpha_i - p$ и α_{i+1} на $\alpha_{i+1} + 1$, тем самым не изменив конечную сумму, но использовав меньше монет.
Поэтому можно воспользоваться жадным алгоритмом: начиная с самого большего номинала брать как можно больше монет. Временная сложность $O(N)$, пространственная $O(N)$.

В случае, если бы номиналы монет были бы произвольны, тогда жадный алгоритм не сработал бы, например при номиналах $1, 3, 4$ и значением $M = 7$ жадный алгоритм предложил бы выбрать 3 монеты номинала 1 и одну монету номинала 4, но есть лучшее решение: взять 2 монеты номинала 3 и одну монету номинала 1. Для решения такой задачи можно использовать алгоритм динамического программирования, который будет иметь временную сложность $O(N * M)$ и пространственную сложность $O(M)$.

Описание программы

Ввиду простоты в реализации лабораторная работа состоит из всего одного файла **main.cpp**.

Тест производительности

Сравнивать время работы алгоритма логично с алгоритмом динамического программирования.

N	P	M	Динамическое программирование	Жадный алгоритм
10	2	10^5	21 ms	7 ms
10	2	10^8	12352 ms	5 ms
100	3	10^8	26666 ms	6 ms

Как видно из таблицы, динамическое программирование начинает сильно уступать жадному подходу при увеличении M, это происходит ввиду того, что M влияет на скорость работы динамического программирования, но не имеет никакого значения в жадном алгоритме.

Выводы

В данной лабораторной работе я познакомился с жадными алгоритмами. Научился проверять задачи на возможность использования жадного подхода. Сравнил динамическое программирование и жадный алгоритм на примере одной задачи.