Московский авиационный институт (национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №7 по курсу «Дискретный анализ»

Студент: К. М. Воронов
Преподаватель: А. А. Кухтичев
Группа: М8О-207Б-19

Дата:
Оценка:
Подпись:

Москва, 2021

Лабораторная работа №7

Задача: При помощи метода динамического программирования разработать алгоритм решения задачи, определяемой своим вариантом; оценить время выполнения алгоритма и объем затрачиваемой оперативной памяти. Перед выполнением задания необходимо обосновать применимость метода динамического программирования.

Разработать программу на языке C или C++, реализующую построенный алгоритм. Формат входных и выходных данных описан в варианте задания:

Имеется натуральное число n. За один ход с ним можно произвести следующие действия: вычесть единицу, разделить на два, разделить на три. При этом стоимость каждой операции — текущее значение n. Стоимость преобразования — суммарная стоимость всех операций в преобразовании. Вам необходимо с помощью последовательностей указанных операций преобразовать число n в единицу таким образом, чтобы стоимость преобразования была наименьшей. Делить можно только нацело.

Формат входных данных

В первой строке строке задано $2 \leqslant n \leqslant 10^7$.

Формат результата

Выведите на первой строке искомую наименьшую стоимость. Во второй строке должна содержаться последовательность операций. Если было произведено деление на 2 или на 3, выведите /2 (или /3). Если же было вычитание, выведите -1. Все операции выводите разделяя пробелом.

1 Описание

Требуется решить задачу с помощью динамического программирования.

Как сказано в [1]: «Динамическое программирование — способ решения сложных задач путём разбиения их на более простые подзадачи.» Также, при решении некоторых задач с использованием динамического программирования характерно использовать уже посчитанные данные.

Чтобы решить свою задачу, я создал массив dp, где индекс обозначает число, а в самом массиве хранится минимальная стоимость операций для получения этого числа. Чтобы заполнить этот массив, я иду сначала, записывая в dp[1] ноль, так как нам нужно прийти именно туда. Далее я начинаю смотреть на индексы $i + 1$, $i * 2$ и $i * 3$. Так как мне надо взять минимальную стоимость, а я иду сначала, то я в каждую ячейку записываю максимум от текущего значения и уже записанного там. В итоге, последовательно решая таким образом мелкие задачи, находя минимальную стоимоть получения чисел до искомого, решаю задачу. По мере прохождения я также записываю в отдельный массив op сами операции для каждого индекса.

2 Исходный код

```cpp
#include <iostream>
#include <vector>

using namespace std;

int main() {
    int n;
    cin >> n;
    vector<long long> dp(n + 1, -1);
    vector<int> op(n + 1);
    dp[1] = 0;
    for (int i = 1; i < n; ++i) {
        if (dp[i + 1] == -1 || dp[i + 1] > dp[i] + i + 1) {
            dp[i + 1] = dp[i] + i + 1;
            op[i + 1] = 1;
        }
        if (i * 2 <= n && (dp[i * 2] == -1 || dp[i * 2] > dp[i] + i * 2)) {
            dp[i * 2] = dp[i] + i * 2;
            op[i * 2] = 2;
        }
        if (i * 3 <= n && (dp[i * 3] == -1 || dp[i * 3] > dp[i] + i * 3)) {
            dp[i * 3] = dp[i] + i * 3;
            op[i * 3] = 3;
        }
    }
    printf("%lld\n", dp[n]);
    int i = n;
    while(i > 1) {
        if (op[i] == 1) {
            i = i - 1;
            printf("-1 ");
        }

        if (op[i] == 2) {
            i = i / 2;
            printf("/2 ");
        }

        if (op[i] == 3) {
            i = i / 3;
            printf("/3 ");
        }
    }
    printf("\n");
}
```

3 Консоль

```
kirill@kirill-G3-3779:~/DA/lab7$ ./s*
10
21
/2 -1 /2 -1
kirill@kirill-G3-3779:~/DA/lab7$ ./s*
21
36
/3 -1 /3 -1
```

4 Тест производительности

Тест производительности представляет из себя следующее: сравнение жадного алгоритма и алгоритма с использованием динамического программирования. В качестве тестов возьмём два числа: 10000, 100000.

```
kirill@kirill-G3-3779:~/DA/lab7$ ./s*
10000
Динамическое программирование: 0.371ms
kirill@kirill-G3-3779:~/DA/lab7$ ./a.out
10000
Жадный алгоритм: 0.004ms
```

```
kirill@kirill-G3-3779:~/DA/lab7$ ./s*
100000
Динамическое программирование: 2.763ms
kirill@kirill-G3-3779:~/DA/lab7$ ./a.out
100000
Жадный алгоритм: 0.004ms
```

Так как при жадном алгоритме не создаётся вектор, он работает быстрее, но не гарантирует правильный ответ.

5 Выводы

Выполнив седьмую лабораторную работу по курсу «Дискретный анализ», я вспомнил и применил такой способ решения задач, как динамическое программирование. Данный метод помогает структурировать решение задачи, разбивая её на маленькие, как говрилось выше. Этот подход применим ко многим вещам и вне программирования. Например, постройка дома: идёт разбитие на постройку фундамента, стен, крыши, окон и т.д., а в итоге получается полноценная постройка. Таких примеров можно найти немало, то есть использовать идею этого способа в жизни очень приемлемо.

Список литературы

[1] Динамическое программирование
https://ru.wikipedia.org/wiki/Динамическое_программирование

[2] Лекции Н.К.Макарова, Московский авиационный институт.