Лабораторная работа № 7 по курсу дискретного анализа: Динамическое программирование

Выполнил студент группы М8О-308Б-20 МАИ *Зинин Владислав*.

Условие

1. 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.

2. **Вариант 4: Игра с числом** 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.

Метод решения

Данная задача решается методом динамического программирования, а конкретно, методом восходящего анализа. Разбив нашу проблему на подзадачи, заметим, что минимальная стоимость $p[n]$ преобразования числа n равна $n + min(p[n/2], p[n/3], p[n-1])$, при этом стоимость $p[n/2]$, $p[n/3]$ считать бесконечной, если n не делится на 2 или на 3 соответственно, так как деление происходит только в нацело. Таким образом, соответствуя принципу восходящего анализа, мы считаем оптимальные ответы для чисел от 2 до n (для числа 1 решение будет равно 0) с сохранением результата в массив, где позиция i соответствует конкретному числу, а также записываем оптимальную операцию во второй вспомогательный массив, где позиция i также соответствует числу.

Описание программы

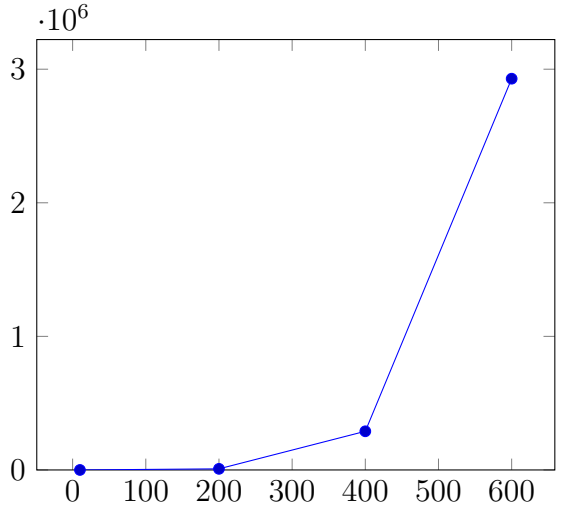
Программа состоит из одного файла.

Дневник отладки

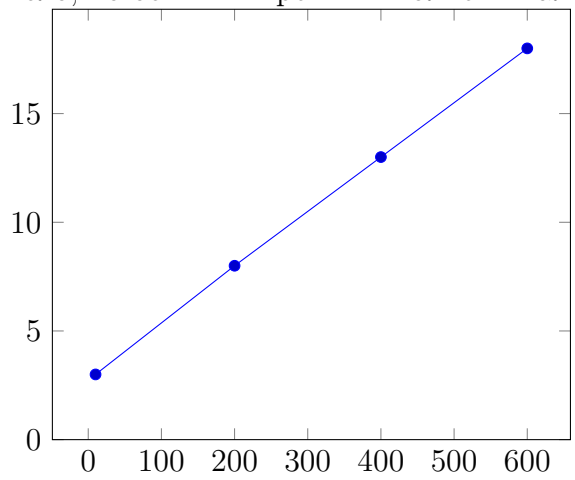
1. Программа была успешно выполнена с первого раза

Тест производительности

Для анализа производительности была написана программа поиска минимальной стоимости преобразования числа n в 1 наивным способом. Ниже приведен тест времени работы наивного алгоритма. По оси X — число, по оси Y — время выполнения алгоритма в мс.



Ниже приведен тест времени работы оптимального алгоритма алгоритма. По оси X — число, по оси Y — время выполнения алгоритма в мс.



Таким образом, мы видим, что при очень маленьких значениях наивный алгоритм работает быстрее оптимального, однако если мы начнем увеличивать входное число, то наивный алгоритм начнет сильно проигрывать, становясь очень медленным.

Выводы

В результате проведенной лабораторной работы я познакомился с решением задач методом динамического программирования, а именно методом восходящего анализа. Подобное решение задач методом разбиения на более мелкие подзадачи дает огромный

выигрышь в производительности в отличие от наивного алгоритма, а значит такой подход для решения многих сложных задач является очень эффективным.