Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №8 по курсу Дискретный анализ

Студент: А. Н. Марков
Преподаватель: Н. А. Зацепин
Группа: М8О-308Б
Дата:
Оценка:
Подпись:

Москва, 2020

Условие

1. Общая постановка задачи

   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++, соответсвующую построенному алгоритму. Формат входных и выходных данных описан в варианте задания.

2. Вариант задания

   Вариант 5. Дана последовательность длины N из целых чисел 1, 2, 3. Необходимо найти минимальное количество обменов элементов последовательности, в результате которых последовательность стала бы отсортированной.

3. Формат входных данных

   Число N на первой строке и N чисел на второй строке.

4. Формат результата

   Минимальное количество обменов.

Метод решения

В жадном алгоритме всегда делается выбор, который кажется самым лучшим в данный момент, т.е. выполняется локально оптимальный выбор, который в конечном итоге приведет к оптимальному решению глобальной задачи.

Алгоритм решения моей задачи: при считывании последовательности будем подсчитывать количество единиц count1 и двоек count2, чтобы сначала расставить единички, а затем двойки на нужные места. Затем будем пробегать с начала последовательности. Первые count1 итераций будем обменивать двойки/тройки с единицами.

- Если обменивается двойка, то будем искать первую встретившуюся единицу, начиная с count1-го индекса последовательности. Как только найдем единицу, обмениваем ее с двойкой.

- Если обменивается тройка, то аналогично с обменом двойки ищем единицу, но теперь с начиная с правого конца последовательности. Это нужно для того, чтобы тройка не занимала место для двоек, тем самым игнорируются лишние обмены.

Таким образом все count1 единиц будут занимать позиции от 0 до $count1 - 1$, а оставшаяся часть последовательности будет состоять только из двоек и троек.

Затем будем рассматривать последовательность [count1, count1 + count2 - 1]. В этой области должны находиться count2 двоек, поэтому нам потребуется только count2 итераций, чтобы обменять тройки в этой области на двойки, находящиеся вне этой области.

После того, как все count2 двоек встанут на свои места, алгоритм завершается, поскольку оставшиеся N - count1 - count2 элементов будут являться тройками, находящимися на правильных местах.

Временная сложность алгоритма $O(N^2)$. Сложность по памяти $O(N)$.

Описание программы

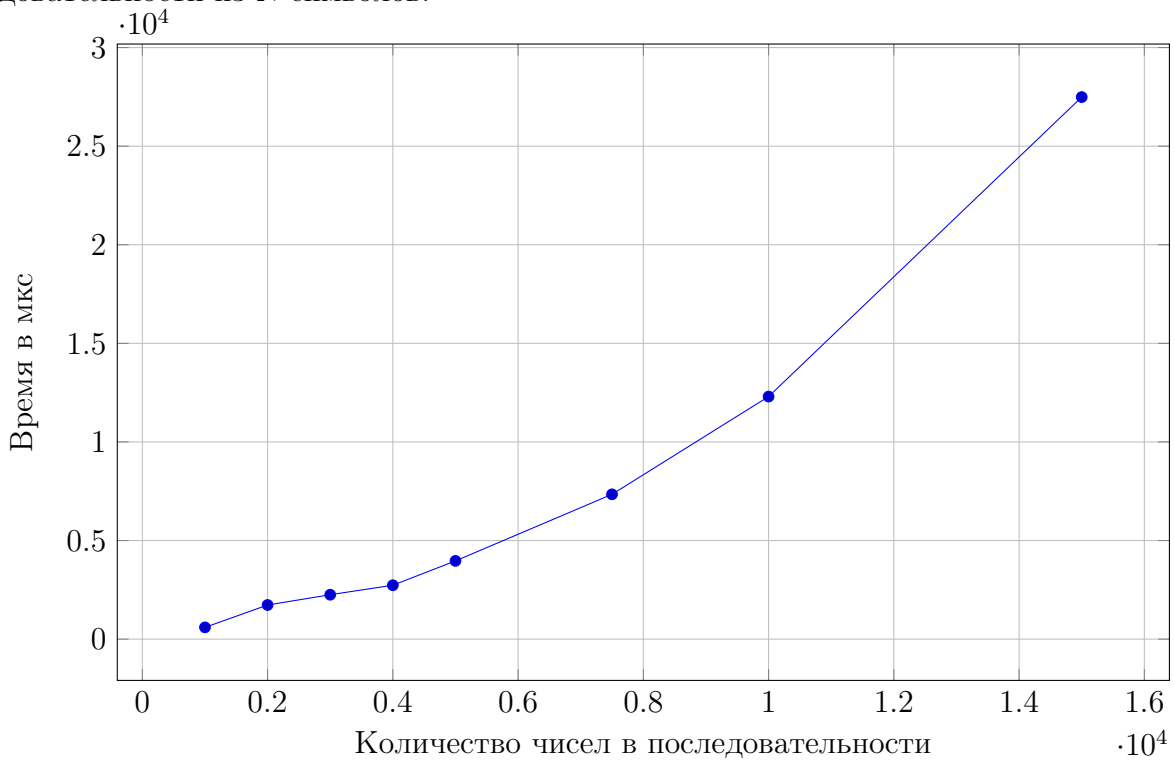
Программа состоит из одного файла main.cpp, в котором находится реализация алгоритма.

Дневник отладки

Программа зашла на чекер с первой попытки.

Тест производительности

Тесты создавались с помощью небольших программы generator.py. Создавались последовательности из N символов.



Выводы

Жадное программирование удобно в задачах, в которых на каждом шаге нужно делать локально наилучший выбор в надежде, что итоговое решение будет оптимальным.

Разница между динамическим программированием и жадными алгоритмами заключается в том, что в динамическом программировании нужно рассматривать несколько решений в поисках оптимального, а в жадных алгоритмах будет только одно предположительно оптимальное решение. Однако такая простота применения в сложных задачах выливается в сложное доказательство применимости жадного алгоритма для ее решения.