Амортизационная стоимость поиска элемента в перестановке из N элементов

Дан массив, содержащий перестановку чисел от 0 до N. Посчитать амортизационную стоимость линейного поиска элемента со значением i. Провести численный эксперимент.

0.1. Описание теории

Амортизационный анализ (англ. *amortized analysis*) – метод подсчёта времени, требуемого для выполнения последовательности операций над структурой данных. При этом время усредняется по всем выполняемым операциям, и анализируется средняя производительность операций в худшем случае.

Средняя амортизационная стоимость операций – величина a, находящаяся по формуле: $a = \frac{\sum\limits_{i=1}^{n} t_i}{n}$, где $t_1, t_2, \ldots, t_n$ – время выполнения операций $1, 2, \ldots, n$, совершённых над структурой данных.

Амортизационный анализ использует следующие методы:

1) Метод усреднения
2) Метод потенциалов
3) Метод предоплаты

Воспользуемся методом усреднения. В этом методе амортизационная стоимость операций определяется напрямую по формуле, указанной выше: суммарная стоимость всех операций алгоритма делится на их количество.

Пусть в перестановке n элементов, тогда время поиска i-го элемента составит $O(n)$. Пусть было произведено m операций, тогда:

$$a = \frac{\sum\limits_{i=1}^{m} t_i}{m} = \frac{m \cdot O(n)}{m} = O(n),$$

где $t_i = O(n)$. Получили: $a = O(n)$

0.2. Эксперимент

Для проведения эксперимента была написана программа на языке C++ (текст программы расположен в файле **st076569_7.cpp** в папке **source**). Она генерирует случайную перестановку из n элементов. Далее n раз случайным образом выбирает $i \in \{0, \ldots, n-1\}$ (значение элемента, который будем искать) и осуществляет поиск, то есть считает кол-во шагов (каждый шаг выполняется за константное время) от начала массива до i. Все значения суммируются и делятся на n. Далее это повторяется еще для двух перестановок. В итоге получаем среднее время поиска элемента (в шагах) по трем случайным перестановкам из n элементов. Обозначим это число как $average_n$.

Эти операции выполняются для $n \in \{200, 400, 600, \ldots, 4000\}$ и для каждого программа возвращает $average_n$. Например для $n = 600$ при нескольких запусках значения $average_{600}$ будут такими: 301.530; 291.667; 292.502. Так же программа для каждого n выводит коэффициент:

$$k = \frac{average_n}{average_{200}}.$$

Он предназначен для анализа пропорциональности роста.

Далее производился анализ того, как меняется $average_n$ в зависимости от n. По теоретическим расчетам зависимость должна быть линейной.

0.3. Результаты и выводы

Программа запускалась 10 раз. Ее вывод перенаправлялся в файл **results.txt** в папке **binaries**. Результаты эксперимента показали, что зависимость времени поиска, выраженного в условных единицах, от количества элементов перестановки с некоторой погрешностью линейна.

Теоретические расчеты были подтверждены эксперементально, амортизационная стоимость линейного поиска элемента со значением i в перестановке чисел от 0 до n равна O(n).