SPb HSE, ПАДИИ, 1 курс, весна 2023/24
Практика по алгоритмам #11

Хеш-таблицы, Random, STL

9 апреля

Собрано 11 апреля 2024 г. в 11:19

Содержание

Хеш-таблицы, Random, STL

1. **STL**

   Дан массив a из n чисел, нужно научиться обрабатывать запросы двух типов.

   - `change(i, y)` — сделать a_i равным y.
   - `intget()` — вернуть индекс i такой, что a_i встречается в массиве наименьшее возможное число раз. Если таких i несколько, вернуть минимально возможный.

2. **Вероятности**

   a) Сколько в среднем нужно сделать просмотров, чтобы найти пустую ячейку в хеш-таблице с открытой адресацией? Например, если $\frac{2}{3}n$ из n ячеек уже заполнено?
   b) Пусть в изначально пустую хеш-таблицу с открытой адресацией размера n мы добавляем n элементов. Оцените матожидание времени работы всего процесса.
   c) Пусть наш алгоритм вероятностный и работает с вероятностью $p = \frac{1}{n}$ корректно. Как получить из него алгоритм с вероятностью ошибки $\frac{1}{10^6}$?
   d) Пример алгоритма поиска делителей числа n: выбрать случайное d, попробовать $gcd(d, n)$. Оцените вероятность ошибки, понизьте ошибку, оцените время работы.

3. **Амортизация в векторе**

   Как работает вектор? Докажите время работы `push_back` вектора $\mathcal{O}(1)$.

4. **Амортизация в хеш-таблице**

   Что делать с хеш-таблицами, если закончилоась свободное место,
   чтобы амортизированное время всех операций осталось $\mathcal{O}(1)$?

5. **Много хеш-таблиц**

   В графе в каждый момент времени $\leqslant m$ рёбер, n вершин. Предложите структуру данных, чтобы быстро добавлять/удалять рёбра и перебирать для вершины соседей.

6. **Анти хеш-тесты для хеш-таблицы на списках**

   a) Предложите плохой тест для хеш-функции $x \to x \bmod p$
   b) Предложите плохой тест для хеш-функции $x \to (ax + b) \bmod p$
   c) Предложите плохой тест для хеш-функции $x \to (ax^2 + bx + c) \bmod p$

7. **(*) Kirkpatrick-sort**

   Хотим отсортировать массив целых чисел от 0 до $2^k - 1$. Идея:

   - Разбить k-битные числа на две половинки по $\frac{k}{2}$ бит.
   - Рекурсивным вызовом отсортировать первые половины.
   - Рекурсивным вызовом отсортировать вторые половины для каждой первой половины.

   Предложите реализацию. Оцените время работы. Причём здесь хеш-таблицы?

Разбор задач практики

1. **STL**

   У задачи есть несколько решений, здесь описано лишь одно из них.

   - a. Храним сам массив $a[i]$.
   - ind. Для каждого x храним множество всех индексов i: $a[i] = x$.
     У множества $ind[x]$ есть `.size()` – сколько раз x встречается.
   - q. При изменениях a обновляем ind, добавляем пару $\langle ind[x].size(), ind[x].min() \rangle$ в множество q кандидатов на ответ.
   - $q.min()$ – ответ на задачу.

2. **Вероятности**

   a) $p = \frac{m}{n}$ – коэффициент заполненности. $E = 1 + pE$ – матожидание числа потыкиваний хеш-таблицы. Решаем линейное уравнение, получаем $E = \frac{1}{1-p}$, для $p = \frac{2}{3}$ получаем 3.

   b) Из успеха $p = \frac{1}{n}$ можно за n повторений получить ошибку $(1 - \frac{1}{n})^n \approx e^{-1}$. Далее повторить ещё $\ln 10^6$ раз. Итого: $n \cdot 20 \cdot \ln 2$.

   c) Самый маленький делитель $x \leqslant \sqrt{n}$, мы попадём в $d \vdots x$ с вероятностью $\frac{1}{x} \Rightarrow gcd(d, n)$ с вероятностью $\geqslant \frac{1}{\sqrt{n}}$ содержит делитель. Повторим $\sqrt{n}$ раз, получим алгоритм за $\sqrt{n} \cdot \log n$ времени и ошибкой $\leqslant e^{-1}$.

3. **Амортизация в векторе**

   Выделили память с запасом (реально хранится $n \leqslant size$ элементов).
   Когда память кончилась ($n = size$), выделили в 2 раза больше, скопировали вектор в новое место. n быстрых операций, затем 1 медленная, среднее время работы $\frac{1}{n+1}(1 + \cdots + 1 + n) \leqslant 2$.

4. **Амортизация в хеш-таблице**

   Как и в векторе, удваиваем память, выделяем h_2. Элементы старой хеш-таблицы h_1 нужно перенести в новую h_2 через функцию `add`: `for` $x \in h_1$`:` hg_2`.add(`x`)`

5. **Много хеш-таблиц**

   Медленное решение: для каждой вершины хеш-таблица номеров соседей.

   Ускорение: хранить одну большую хеш-таблицу пар $\langle a, b \rangle$: ребро $a \to b$.

   Почему быстро? Не будет постоянных удвоений.

6. **Анти хеш-тесты для хеш-таблицы на списках**

   Для всех пунктов: взять $\{0, p, 2p, \dots\}$. Если p фиксировано, всегда есть контртест.

   a) Предложите плохой тест для хеш-функции $x \to x \bmod p$
   b) Предложите плохой тест для хеш-функции $x \to (ax + b) \bmod p$
   c) Предложите плохой тест для хеш-функции $x \to (ax^2 + bx + c) \bmod p$

7. `(*)` **Kirkpatrick-sort**

   См. конспект ПМИ.

Домашнее задание

1. **(2) Пара элементов с заданной суммой**

   Дан массив из n целых чисел от 0 до 10^{18} и целое число S.

   Найдите за $\mathcal{O}(n)$ пару элементов в массиве, сумма которых равна S.

2. **(2) Понижение вероятности**

   Пусть у нас есть алгоритм A с односторонней ошибкой. Алгоритм A работает за $\mathcal{O}(n^3)$, вероятность **успеха** $1/n^3$. Модифицируйте алгоритм, чтобы получить вероятность успеха $1 - 1/2^n$. За сколько теперь работает алгоритм?

3.1. Дополнительная часть

1. **(3) Грустная хеш-таблица**

   Пусть у нас есть идеальная хеш-функция, которая равномерно раскидала n элементов по n спискам хеш-таблицы на списках. Докажите, что матожидание длины максимального из списков есть $\Theta(\log n)$.