Амортизационный анализ

Булгаков Илья, Валерий Сенотов, Виталий Ерошин

Московский физико-технический институт

Москва, 2022

Содержание

Амортизационный анализ

До текущего момента мы привыкли оценивать сложность алгоритмов находя асимтотику худшего сценария.

Однако, бывают случаи, когда такого анализа недостаточно, и можно оценить гараздо лучше!

Амортизационный анализ

Пусть у нас есть некоторая операция `f(n)`. И ее время выполнения зависит от аргумента:

$$\begin{cases} Time(\texttt{f(n)}) = O(n), \text{ если } n = 2^k \\ Time(\texttt{f(n)}) = O(1), \text{ иначе} \end{cases}$$

Как оценить время алгоритма, который запускает `f(n)` на всех `n` от 1 до M?

Амортизационный анализ

Пусть у нас есть некоторая операция `f(n)`. И ее время выполнения зависит от аргумента:

$$\begin{cases} Time(\texttt{f(n)}) = O(n), \text{ если } n = 2^k \\ Time(\texttt{f(n)}) = O(1), \text{ иначе} \end{cases}$$

Как оценить время алгоритма $g(M)$, который запускает `f(n)` на всех `n` от 1 до M?

1. С одной стороны, можно сказать, что асимптотика `f(n)` - $O(n)$, потому что таков худший случай. Тогда $g(M) \sim O(n^2)$

Амортизационный анализ

Пусть у нас есть некоторая операция `f(n)`. И ее время выполнения зависит от аргумента:

$$\begin{cases} Time(\texttt{f(n)}) = O(n), \text{ если } n = 2^k \\ Time(\texttt{f(n)}) = O(1), \text{ иначе} \end{cases}$$

Как оценить время алгоритма $g(M)$, который запускает `f(n)` на всех `n` от 1 до M?

1. С одной стороны, можно сказать, что асимптотика `f(n)` - $O(n)$, потому что таков худший случай. Тогда $g(M) \sim O(n^2)$
2. С другой стороны, можно оценить суммой ряда и получить $g(M) \sim O(n)$

Амортизационный анализ

А теперь у нас есть некоторая структура данных `S`, имеющая метод f(x). И время выполнения `f(x)` зависит уже не от аргумента, а от состояния структуры `S`.

Как оценить время работы `S.f(x)`?

Амортизационный анализ

А теперь у нас есть некоторая структура данных `S`, имеющая метод f(x). И время выполнения `f(x)` зависит уже не от аргумента, а от состояния структуры `S`.

Как оценить время работы `S.f(x)`?

1. Можно перебрать все возможные состояния структуры `S` и рассмотреть худший случай. Но это грубая оценка!

Амортизационный анализ

А теперь у нас есть некоторая структура данных `S`, имеющая метод f(x). И время выполнения `f(x)` зависит уже не от аргумента, а от состояния структуры `S`.

Как оценить время работы `S.f(x)`?

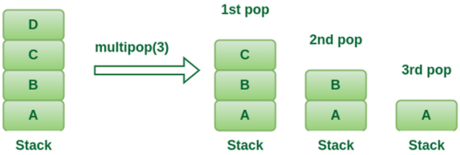
1. Можно перебрать все возможные состояния структуры `S` и рассмотреть худший случай. Но это грубая оценка!
2. А можно воспользоваться методом из амортизационного анализа и получить амортизированное время работы.

Метод усреднения

$\{op_1(ds), op_2(ds), ..., op_n(ds)\}$ — последовательность операций над структурой данных.
t_j — время выполнения операции $op_j(ds)$.
$a = \frac{\sum_{j=1}^{n} t_j}{n}$ — средняя амортизационная стоимость операций.

Метод усреднения

Пример



В стек добавлена операция multipop(). Найти амортизационную сложность всех операций такого стека, в который было добавлено m элементов.

$a = \frac{\sum_{j=1}^{n} t_j}{n} = \frac{\sum_{j=1}^{n} \sum_{i=1}^{m} t_{ji}}{n} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} t_{ji}}{n}$

Каждый элемент мог быть не более одного раза добавлен и удален.

$\implies \sum_{j=1}^{n} t_{ji} \leq 2$

$a = O(\frac{\sum_{i=1}^{m} 2}{n}) = O(\frac{2m}{n}) = O(1)$

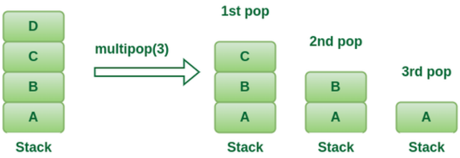
Метод потенциалов

Введём для каждого состояния структуры данных величину Φ — потенциал. Изначально потенциал равен Φ_0, а после выполнения i-й операции — Φ_i. Стоимость i-й операции обозначим $a_i = t_i + \Phi_i - \Phi_{i-1}$.
Пусть n — количество операций, m — размер структуры данных. Тогда средняя амортизационная стоимость операций $a = O(f(n, m))$, если выполнены два условия:

1. $\forall i : a_i = O(f(n, m))$
2. $\forall i : \Phi_i = O(nf(n, m))$

Метод потенциалов

Пример



Пусть потенциал — это количество элементов в стеке.

1.
   - $a_{push} = 1 + 1 = 2$
   - $a_{pop} = 1 - 1 = 0$
   - $a_{multipop} = k - k = 0$
2. $\forall i : \Phi_i = O(n)$ (количество элементов не больше n)

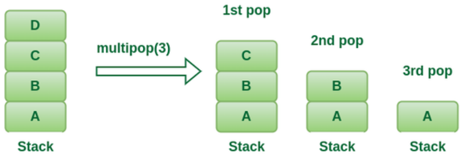
$f(n, m) = 1, a = O(1)$

Метод монет

c_j — учётная стоимость операции (в монетах).
R — резерв монет, в начале операций равен 0.
$R_j = R_{j-1} + (c_j - t_j)$ — формула изменения резерва после выполенения операции.
$\forall j : c_j = O(f(n, m)), R_j \geq 0 \implies a = O(f(n, m))$

Метод монет

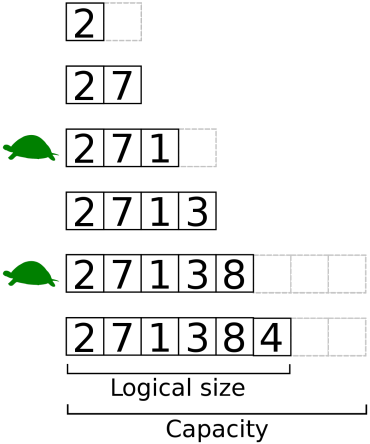
Пример



- $c_{push} = 2 = O(1), R+ = 1$
- $c_{pop} = 0 = O(1), R- = 1$
- $c_{multipop} = 0 = O(1), R- = k$

$a = O(1)$

Динамический массив

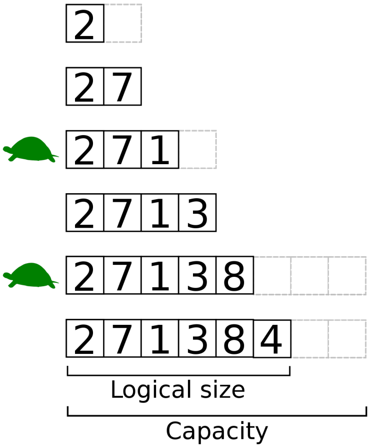


- Введем два аттрибута нашего массива - `size` и `capacity`
- Храним статический массив размера `capacity`
- Когда приходит запрос `push_back`, добавляем элемент в наш массив
- Если `size` стал равен `capacity`, увеличиваем `capacity` в 2 раза, создаем новый статический массив, перекладываем элементы из старого массива в новый

Какая асимптотика у `push_back`?

Динамический массив

Асимптотика push_back



- Что будет, если мы добавим n элементов?
- Посчитаем количество операций, добавляющих элемент в массив. Их всего n. Сложность: $nO(1) = O(n)$
- Посчитаем количество операций, затраченных на расширение массива:

$$2+4+\ldots+O(\frac{n}{2})+O(n) < 3n = O(n)$$

- Суммарная сложность $O(n)$, операций n. Значит амортизационная сложность (по методу усреднения) - $O^*(1)$

Динамический массив

Асимптотика pop_back

Что можно сказать про асимтотику pop_back?

Hint: идея аналогичная.

Полезные ссылки I

- Т.Кормен, Ч.Лейзерсон, Р.Ривест, К.Штайн - Алгоритмы. Построение и анализ. Глава 17
https://bit.ly/2wFzphU
- Амортизационный анализ
https://neerc.ifmo.ru/wiki/index.php?title=Амортизационный