Amortized Analysis

CLRS Reading: Sections 17.1, 17.2, 17.3, pages 451 – 463

Stack Operations*

*Each operation runs in $O(1)$ time.
MultiPop($S$, $k$)

MultiPop($S$, $k$)
while not Stack-Empty($S$) and $k \neq 0$
Pop($S$)
$k = k-1$

Worst-Case Time

• In a sequence of $n$ stack operations starting with an empty stack, what is the worst-case cost of a multiPop operation?

• Worst-case cost of $n$ stack operations?
Aggregate Method*

*Each object can be popped at most once for each time it is pushed.

Aggregate Method

**Actual Costs**
- Push(S, A) \(\mathcal{O}(1)\)
- Pop(S) \(\mathcal{O}(1)\)
- MultiPop(S, k) \(\mathcal{O}(\text{min}(k, \text{size}(S)))\)

**Amortized Costs:**
- All \(T(n)/n = \mathcal{O}(n)/n = \mathcal{O}(1)\)
Accounting Method

**Actual Costs**
- \(\text{Push}(S, A)\): $1
- \(\text{Pop}(S)\): $1
- \(\text{MultiPop}(S, k)\): $\min(k, \text{size}(S))

**Amortized Costs**
- \(\text{Push}(S, A)\): $2
- \(\text{Pop}(S)\): $0
- \(\text{MultiPop}(S, k)\): $0

Money in the Bank*

*No bounced checks.*
Potential Method

- Associate a potential energy with the data structure at each stage.

\[ \hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) \]

Total Amortized Cost

- \[ \sum_{1 \leq i \leq n} \hat{c}_i = \sum_{1 \leq i \leq n} (c_i + \Phi(D_i) - \Phi(D_{i-1})) \]
  \[ = c_n + \Phi(D_n) - \Phi(D_{n-1}) + \]
  \[ = c_{n-1} + \Phi(D_{n-1}) - \Phi(D_{n-2}) + \]
  \[ = c_{n-2} + \Phi(D_{n-2}) - \Phi(D_{n-3}) + ... + \]
  \[ = c_1 + \Phi(D_1) - \Phi(D_0) \]
  \[ = \sum_{1 \leq i \leq n} c_i + (\Phi(D_n) - \Phi(D_0)) \]

- Thus, if we define \( \Phi \) so that \( \Phi(D_i) \geq \Phi(D_0) \geq 0 \) for all \( i \), the total amortized cost is an upper bound on the total cost.
For Example

\[ \Phi = \text{size of } S \]

1. Empty Stack
2. Push(S, A)
3. Push(S, B)
4. Push(S, C)
5. MultiPop(S, 2)
6. Push(S, B)
7. MultiPop(S, 5)

Ripple Counter

3 2 1 0
Software Ripple Counter*

Increment \((A)\)
\[
i = 0 \\
\text{while } i < A.length \text{ and } A[i] = 1 \\
\quad A[i] = 0 \\
\quad i = i+1 \\
\text{if } i < A.length \\
\quad A[i] = 1
\]

*Cost of an increment is linear in the number of flipped bits.
Total number of flips for a sequence of \(n\) increments?

Aggregate Method

Flips every fourth call
Flips every call
Flips every other call
Accounting Method

\begin{align*}
\text{Increment}(A) \\
& \quad i = 0 \\
& \quad \text{while } i < A\text{.length and } A[i] = 1 \\
& \quad \quad A[i] = 0 \\
& \quad \quad i = i + 1 \\
& \quad \text{if } i < A\text{.length} \\
& \quad \quad A[i] = 1
\end{align*}

Potential Method

- We seek a potential function \( \Phi \) so that \( \Phi(D_i) \geq \Phi(D_0) \geq 0 \).
Potential Method

• Suppose the $i^{th}$ Increment operation resets (sets to 0) $t_i$ bits, so $c_i \leq t_i + 1$. The number of 1’s in the counter after the $i^{th}$ iteration is $b_i \leq b_{i-1} - t_i + 1$. (Why $\leq$?)

• The potential difference is
  \[
  \Phi(D_i) - \Phi(D_{i-1}) = b_i - b_{i-1} \\
  \leq (b_{i-1} - t_i + 1) - b_{i-1} \\
  = 1 - t_i
  \]

• The amortized cost is therefore
  \[
  \hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1}) \\
  \leq (t_i + 1) + (1 - t_i) \\
  = 2
  \]