# Lab07-Amortized Analysis

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2020.

∗ If there is any problem, please contact TA Shuodian Yu.
∗ Name: Futao Wei Student ID: 518021910750 Email: weifutao@sjtu.edu.cn

1. For the TABLE-DELETE Operation in Dynamic Tables, suppose we construct a table by multiplying its size by $\frac{2}{3}$ when the load factor drops below $\frac{1}{3}$. Using *Potential Method* to prove that the amortized cost of a TABLE-DELETE that uses this strategy is bounded above by a constant.

   **Solution.** We start by defining a potential function

   $$\Phi(T) = size[T] - num(T)$$

   **Correctness:** the potential is 0 for an empty table, and $\Phi(T)$ never goes negative. Thus, the total amortized cost of a sequence of $n$ TABLE-DELETE operations with respect to $\Phi$ is an upper bound of the actual cost.
   **Case 1:** no contraction

   $$\begin{aligned}\hat{C}_i &= C_i + \Phi_i - \Phi_{i-1} \\ &= 1 + (size_i - num_i) - (size_{i-1} - num_{i-1}) \\ &= 2\end{aligned}$$

   **Case 2:** a contraction is triggered

   $$\begin{aligned}\hat{C}_i &= C_i + \Phi_i - \Phi_{i-1} \\ &= num_i + 1 + (size_i - num_i) - (size_{i-1} - num_{i-1}) \\ &= num_{i-1} + 1 - \frac{1}{3} size_{i-1} \\ &= 1\end{aligned}$$

   Hence the amortized cost of a TABLE-DELETE that uses this strategy is bounded above by a constant. □

2. A **multistack** consists of an infinite series of stacks $S_0, S_1, S_2, \cdots$, where the $i^{th}$ stack $S_i$ can hold up to $3^i$ elements. Whenever a user attempts to push an element onto any full stack $S_i$, we first pop all the elements off $S_i$ and push them onto stack $S_{i+1}$ to make room. (Thus, if $S_{i+1}$ is already full, we first recursively move all its members to $S_{i+2}$ .) An illustrative example is shown in Figure 1. Moving a single element from one stack to the next takes $O(1)$ time. If we push a new element, we always intend to push it in stack $S_0$.

   (a) In the worst case, how long does it take to push a new element onto a multistack containing $n$ elements?

   (b) Prove that the amortized cost of a push operation is $O(\log n)$ by *Aggregation Analysis*.

   (c) (Optional Subquestion with Bonus) Prove that the amortized cost of a push operation is $O(\log n)$ by *Potential Method*.

   **Solution.**

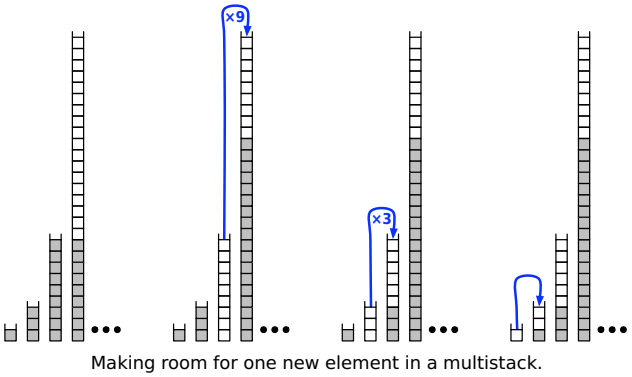


图 1: An example of making room for one new element in a multistack.

(a) In the worst case, $n = \sum_{i=0}^{k} 3^i$, where pushing a new element requires $n$ pops and $n$ pushes first. Hence $T(n) = 2n + 1$.

(b) $S_i$ becomes full every $3^i$ pushes since the first time $S_0, S_1, \cdots, S_{i-1}$ becomes full. Hence the total cost for $S_i$ is

$$\frac{n - \frac{3^i - 1}{2}}{3^i} \cdot 2 \cdot 3^i \sim n, \quad i = 0, 1, \cdots$$

Since there are approximately $\log n$ stacks, we have the aggregate cost $\sim n \log n$. Thus the amortized cost of a push operation is $O(\log n)$.

(c) We start by defining a potential function

$$\Phi(T) = \sum_{i=0}^{k} num_i \cdot (k - i)$$

where $num_i$ stands for the number of elements in $S_i$ and $k$ for the number of non-empty stacks.
Suppose that the elements in $S_0, S_1, \cdots, S_j (j \geq 0)$ are shifted right after pushing a new element, we have

$$C_i \leq \sum_{t=0}^{j} 3^t = \frac{3^{j+1} - 1}{2}$$

Hence the amortized cost of a push operation is

$$\begin{aligned}
\hat{C}_i &= C_i + \Phi_i - \Phi_{i-1} \\
&\leq \frac{3^{j+1} - 1}{2} + \sum_{t=1}^{j} 3^{t-1} \cdot (k - t) + 3^j \cdot (k - j - 1) - \sum_{t=0}^{j} 3^t \cdot (k - t) \\
&\sim k \\
&\sim O(\log n)
\end{aligned}$$

□

3. Given a graph $G = (V, E)$, and let $V'$ be a strict subset of $V$. Prove the following propositions.

(a) Let $T$ be a minimum spanning tree of a $G$. Let $T'$ be the subgraph of $T$ induced by $V'$, and let $G'$ be the subgraph of $G$ induced by $V'$. Then $T'$ is a minimum spanning tree of $G'$ if $T'$ is connected.

(b) Let $e$ be a minimum weight edge which connects $V'$ and $V \setminus V'$. There exists a minimum weight spanning tree which contains e.

**Solution.**

(a) If $T'$ is connected, then $T'$ is obviously a tree.
Suppose for contradiction that $T'$ is not a minimum spanning tree of $G'$ and $S$ is one. In this case, we can get a spanning tree of $G$ by connecting $S$ with $T \setminus T'$, which is smaller than $T$ — we get a contradiction! Hence $T'$ is a minimum spanning tree of $G'$.

(b) Assume for contradiction that no minimum weight spanning tree contains $e$. But any minimum spanning tree has to connect $V'$ and $V \setminus V'$ via a certain edge, say, $e'$. If we replace $e'$ with $e$, we'll get a spanning tree which is no worse than minimum, since $e$ is a minimum weight edge which connects $V'$ and $V \setminus V'$. — A contradiction! The proposition is thus proved.

□

**Remark:** Please include your .pdf, .tex files for uploading with standard file names.