CSC263 Wednesday Lecture — Feb 28, 2018

Recall: **Amortized Analysis**

- $T(n)$: max cost of executing any sequence of n operations
- $A(n) = T(n)/n$: amortized cost of an

Dynamic Tables

Brute Force

$$T(n) \leq n + \sum_{i=0}^{\lfloor \log_2 n \rfloor} 2^i = n + \frac{2^{\lfloor \log 2 \rfloor + 1} - 1}{2 - 1} \leq n + 2^{\lfloor \log n \rfloor + 1}$$

$$T(n) \leq n + 2 \cdot 2^{\lfloor \log n \rfloor} \leq n + 2n = 3n$$

$$\therefore A(n) = T(n)/n \leq 3n/n = 3$$

- Accounting method:
  - charge 1 to place element, 1 to copy element, and 1 to copy associated element in other half of array
  - start by thinking "what is an expensive operation and how can I collect credits to account for it?"
  - INVARIANT: When table is full, you have collected 1 credit for every element in the table

- Table Contraction & Expansion:

a) if $\alpha(T) = 1$ and INSERT occurs
move to new T_2 | $|T_2| = 2 \cdot |T_2|$ // T_2 half empty

b) if $\alpha(T) = 1/4$ and DELETE occurs
move to new T_2 | $|T_2| = 1/2 \cdot |T_1|$ // T_2 half empty

(size n)

- Starting from a half full table, two possible expensive ops triggered:

a) $n/2$ inserts, fills table; $2 \cdot n/2 = n$ credits saved
copy all n into T_2 using n credits

b) $n/4$ deletes; table $1/4$ full; $1 \cdot n/4 = n/4$ credits saved
copy all $n/4$ into T_2 using $n/4$ credits

$\therefore$ charge 3 for INSERT, only 2 for DELETE