Quantitative Hoare Logic

CS536 Science of Programming

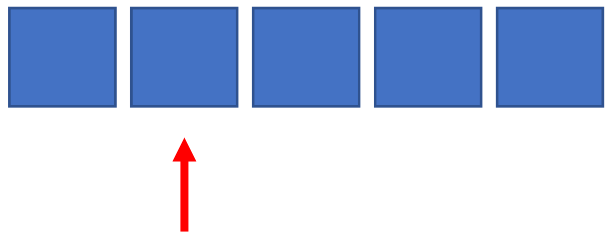
Lecture 25

Quentin Carbonneaux, Jan Hoffmann, Tahina Ramananandro, Zhong Shao. “End-to-End Verification of Stack-Space Bounds for C Programs.”

PLDI 2014

Amortized Analysis Overview

- Build a queue as a linked list
- Enqueue: 1 LL operation
- Dequeue: n LL operations



Amortized Analysis Overview

- Faster: Use 2 linked lists:
- Enqueue still 1 LL op
- Dequeue: 1 LL op
  - Unless the back is empty, then n LL ops

- But: a program that does n enqueues and m (<= n) dequeues will do at most 3n LL ops

Amortized Analysis Overview

- Let Q = 2 x # of elements in “Front” queue
  + 3 x # of enqueues left to do
  + # of elements in “Back” queue
- Q can never be negative
- Q = 3n at start of program
- Enqueue does 1 op, decreases Q by 1
- Dequeue does k + 1 ops, decreases Q by 2k – (k – 1) = k + 1
- => # of ops <= 3n

Amortized Analysis Overview

- Q is a *potential function*
- It's a function of the state σ: $Q(\sigma)$

- Defns:
- $(Q_1 + Q_2)(\sigma) = Q_1(\sigma) + Q_2(\sigma)$
- $k(\sigma) = k$
- $Q_1 \leq Q_2$ iff $Q_1(\sigma) \leq Q_2(\sigma)$ for all σ

Quantitative Hoare Logic

- If $\sigma \vDash \{Q\}\ s\ \{Q'\}$ and $\langle s, \sigma\rangle \to^n \ \langle skip, \sigma'\rangle$, then $n \leq Q(\sigma)$

- Can have other definitions of "cost" as well, not just counting steps

Inference rules

$$\frac{}{\vdash \{Q+1\}\ x := e\ \{Q\}} \qquad \frac{}{\vdash \{Q\}\ \text{skip}\ \{Q\}} \qquad \frac{\vdash \{Q\}\ s_1\ \{Q'\} \qquad \vdash \{Q'\}\ s_2\ \{Q''\}}{\vdash \{Q\}\ s_1; s_2\ \{Q''\}}$$

$$\frac{\vdash \{Q\}\ s_1\ \{Q'\} \qquad \vdash \{Q\}\ s_2\ \{Q'\}}{\vdash \{Q+1\}\ \text{if}\ e\ \text{then}\ \{s_1\}\ \text{else}\ \{s_2\}\ \{Q'\}} \qquad \frac{\vdash \{Q\}\ s\ \{Q\}}{\vdash \{Q+2\}\ \text{while}\ e\ \{s\}\ \{Q\}}$$

$$\frac{\vdash \{Q_1\}\ s\ \{Q_2\} \qquad Q_1 \le Q_1' \qquad Q_2' \le Q_2}{\vdash \{Q_1'\}\ s\ \{Q_2{}'\}}$$

Proof Outlines

$$
\begin{array}{ll}
 & \{2n + 4\} \\
i := 0; & \{2(n - i) + 3\} \\
s := 0; & \{2(n - i) + 2\} \\
\text{while } (i < n)\ \{ & \{2(n - i) = 2\big(n - (i + 1)\big) + 2\} \\
\quad s := s + a[i]; & \{2(n - (i + 1)) + 1\} \\
\quad i := i + 1 & \{2(n - i)\} \\
\} & \{2(n - i)\} \Rightarrow \{0\}
\end{array}
$$

* Let's forget about the skip; s_2 step.