Three common techniques

Communica the average performance of each operation in worst case.

- The amortized method (not discussed today)
- The accounting method
- Aggregate analysis

The time required to perform a sequence of data structure operations is averaged over all the operations performed.

AMORTIZED ANALYSIS

- Accounting method
- Binary Counter Problem
- Amortized analysis
- Lecture Realu
- Post Discussion

6.006 Recitation 7
25 February 2008
sequence of n operations \( \in O(n^2) \) (we can achieve tighter bounds on the worst-case running time of any stack operation \( \in O(n) \))

Consider a sequence of n push, pop, and multipop operations. Stack initially empty.

Abstract costs of push-1, pop-1, multipop(s,k) \( \in \min(s,k) \).

K

while not stack-empty(s) and k > 0

MULTIPOP(s,k)

POP(s): pops the top of stack s and returns the popped object.

PUSH(s,x): pushes x on stack s

Stack Operations

operation is \( \in T(n) \).

Show sequence of n operations takes \( \in T(n) \) time in worst case, amortize cost per

Aggregate Analysis
x = \sum_{i=0}^{k-1} A[i] \cdot 2^i

Binary numbers x stored in A, lowest order bit in A[0] and highest order in A[k-1]

length(A) = k

k-bit binary counter

Incrementing a Binary Counter

Average cost of operation is O(n/ln n) = O(1)

Number of times pop can be called on non-empty stack \leq n

Number of push operations \geq n

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

Sequence of n push, pop, multipop can almost cost O(n)
The Accounting Method

Amortized cost per operation = \( \frac{c(m)}{n} = O(1) \)

For other operations,

Amortized cost \( > \) actual cost.

Assign different changes to different operations (some charged more, some less).
Amount of credit is always non-negative, so number of positive or non-negative
Multipop = no change; again, use the credit

Pop = no change. For operation, pay using the credit.

At any point, every plate has 0 credit on the stack.

Push a plate, a dollar to push and a dollar as credit at top of it.

A dollar bill = a unit of cost

Stack Operations

Stack Operations

Total credit stored in the data structure $3a - \frac{1}{2} a > 0$ at all times.
Amortized Complexity

Dynamic Hash Tables

Amount of credit \( \geq 0 \)

Amount of operations \( \in O(n) \)

Number of bits in the counter \( \geq 0 \)

Amortized cost of 1 increment \( \in O(1) \)

At most one bit is set

While loop cost of resetting bits \( \in O(\text{paid by the credit on the bits}) \)

Increment

Increment to be used if data for flipping back to 0

When a bit is set, use \( $1 \) to pay for additional setting of bit, place \( 0 \) as credit on the bit

\( \$2 \) to set a bit to 1

Amortized cost

Incrementing a binary counter