Amortized Analysis

input size n → [Alg] →

↑ how long in terms of n

Sequence of operations $o_1\ o_2\ o_3 \ldots o_n$

$o_i \in \{\text{operations on a D.S.}\}$

Example B.S.T.
- search
- insert
- delete
- succ
- pred
- ⋮

search, insert → $\Theta(\lg n)$

delete, succ, pred → $\Theta(\lg n)$

Stack
- push
- pop
- MULTIPOP(k)

$O(1)$

n stack ops

$o_1\ o_2 \cdots o_n$ (AMORTIZED COST)

$O(1) \times n = \Theta(n)$

Goal get cost $\frac{1}{n}\sum_{i=1}^{n} \text{cost}(o_i)$ — avg cost

worst case avg cost

Lazy D.S.

- push — $O(1)$
- pop — $O(1)$
- multipop(k) — $\Theta(k)$

$o_1\ o_2\ o_3 \ldots o_n$

worst case operation $\Theta(n)$ × n operations

$\Rightarrow \Theta(n^2)$ ~~$\Theta(n^2)$~~ $\Theta(n)$ cost of sequence

Aggregate $\Theta(n)$

AGGREGATE METHOD

Stack w/ multipop

Accountant's soln — $1 item handled

| $2/push
| pops – free

$2 → $1 copied to item }
$2 → $1 to grad student

$2/push
$0/pop
$0/multipop
} AMORTIZED COSTS

$o_1, o_2 \cdots o_n$

\$2 or \$0

$\Theta(n)$

Banker's soln (Physicist's soln)

keep bank acct w/ \$

total in acct THE POTENTIAL] NON NEGATIVE

Φ(data structure)

Actual cost = Amortized cost + $\Delta\Phi$

Φ(after) – Φ(before)

↑ pay grad student

↑ customer pays

per operation

$$\text{Actual cost} = \text{Amortized cost} + \Phi(\text{after}) - \Phi(\text{before})$$

↑ pay for it — ↑ customer pays — ↑ out of our pocket

Sequence of ops $o_1, o_2 \ldots o_n$

$$\sum_{i=1}^{n} \text{cost}(o_i) = \sum \text{amortized costs} + \Phi(\text{initial potential}) - \Phi(\text{final potential})$$

ACTUAL COST

initial potential = 0; final potential ≥ 0

~~NEGT~~ NON-POSITIVE ≤ 0

$$\frac{1}{n}\left[\text{actual cost sequence}\right] \leq \frac{1}{n}\sum \text{amortized costs}$$

<u>Stack w/ multipop</u> $\Phi(\text{stack}) = \#$ of items in stack

$$\text{AMORTIZED COST} = \text{ACTUAL COST} - \Phi(\text{before}) + \Phi(\text{after})$$

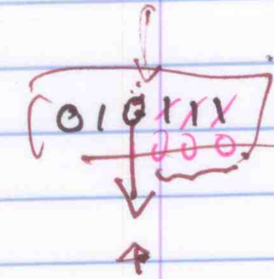
$\Delta\Phi$

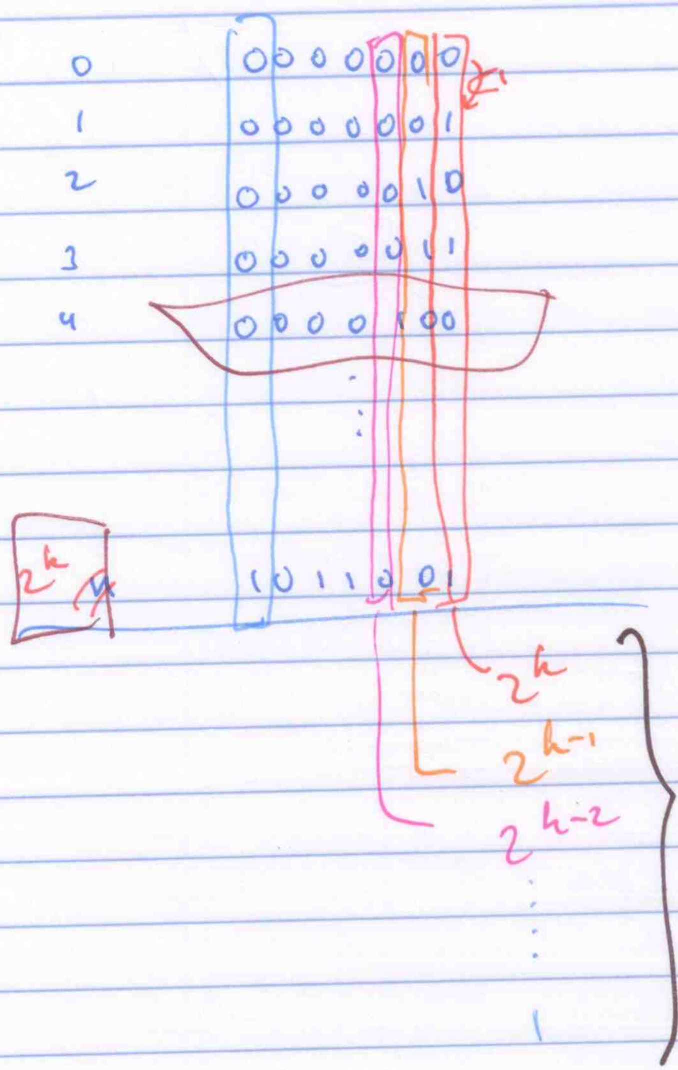
<u>Amort cost</u>

push	\$1	+\$1	= \$2
pop	\$1	−\$1	= \$0
multipop(k)	\$k	−\$k	= \$0

Binary counter

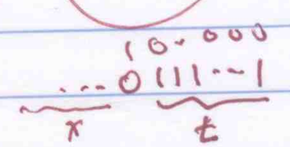
increment counter





$$1+2+4+8+\cdots+2^k = 2^{k+1}-1$$

$2/op.



Φ(counter) = # of 1-bits

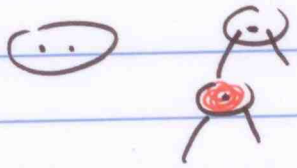
Amortized cost increment = # of bits that change − # of 1 bits before + # 1-bits after

= # 1's at right + 1 − (x + t) + x + 1

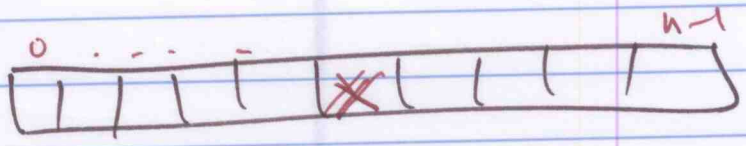
= t + 1 − t + 1 = 2

P.I.D.O.O.M.A

Φ (data structure) — zero when DS in good shape; "large" bad shape



Hash Tables



key → address function

HASH FCN depends on table size