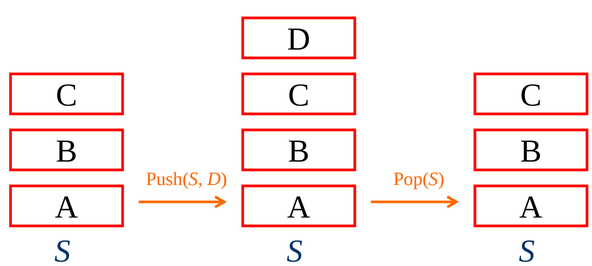
Fundamental Algorithms

Amortized Analysis

CLRS Reading: Sections 17.1, 17.2, 17.3, pages 405 -- 416
Midterm Exam: Friday, April 18

Fundamental Algorithms

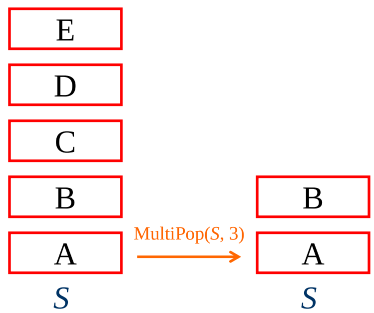
Stack Operations*



*Each operation runs in $O(1)$ time.

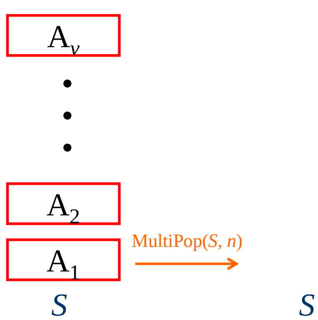
MultiPop(*S*, *k*)

```
MultiPop(S, k)
    while not Stack-Empty(S) and k ≠ 0
        do Pop(S)
           k ← k-1
```



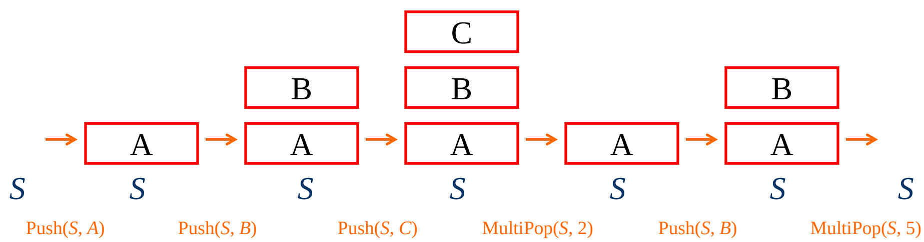
Worst-Case Time

- In a sequence of *n* stack operations starting with an empty stack, what is the worst-case cost of a `multiPop` operation?



- Worst-case cost of *n* stack operations?

Amortized Cost*



*Aggregate Method: Each object can be popped at most once for each time it is pushed.

The Accounting Method

Actual Costs

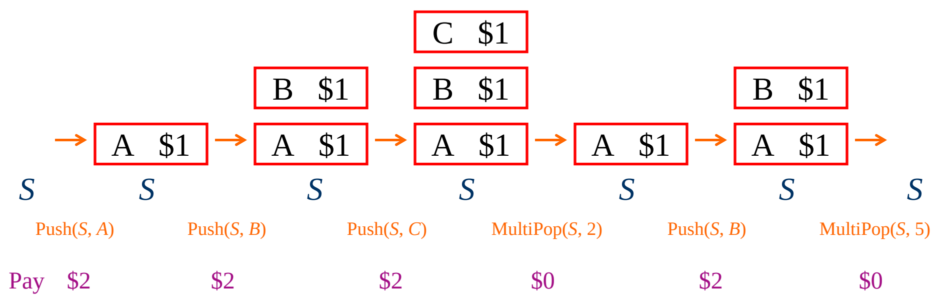
`Push(S, A)`	\$1
`Pop(S)`	\$1
`MultiPop(S, k)`	\$min(*k*, *size*(*S*))

Amortized Costs

`Push(S, A)`	\$2	← Overcharge Peter to pay Paul
`Pop(S)`	\$0	← Paul
`MultiPop(S, k)`	\$0	← Paul

Fundamental Algorithms

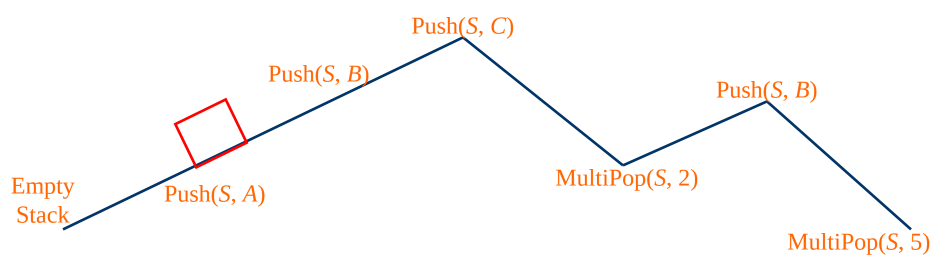
Money in the Bank*



*No bounced checks.

Fundamental Algorithms

Potential Energy



- Associate a potential energy with the data structure at each stage.

Change in potential due to operation

Amortized cost → $\hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1})$

Actual cost

Fundamental Algorithms

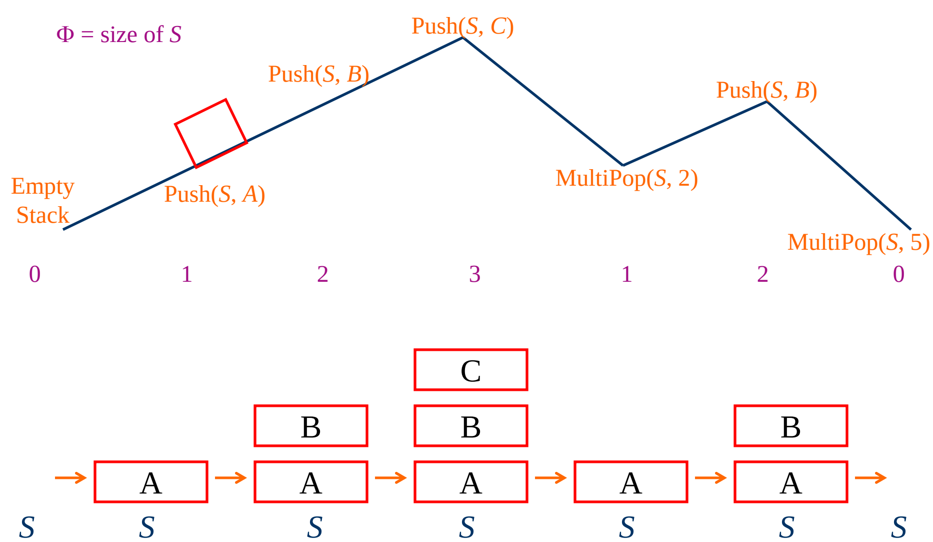
Total Amortized Cost

- $$\begin{aligned}\Sigma_{1\le i\le n}\,\hat{c}_i &= \Sigma_{1\le i\le n}\,(c_i + \Phi(D_i) - \Phi(D_{i-1})) \\ &= c_n + \Phi(D_n) - \Phi(D_{n-1}) + \\ &\quad\; c_{n-1} + \Phi(D_{n-1}) - \Phi(D_{n-2}) + \\ &\quad\; c_{n-2} + \Phi(D_{n-2}) - \Phi(D_{n-3}) + \ldots + \\ &\quad\; c_1 + \Phi(D_1) - \Phi(D_0) \\ &= \Sigma_{1\le i\le n}\, c_i + (\Phi(D_n) - \Phi(D_0))\end{aligned}$$
- Thus, if we define Φ so that $\Phi(D_i) \ge \Phi(D_0) \ge 0$ for all i, the total amortized cost is an upper bound on the total cost.

S - 9

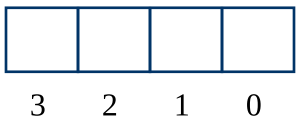
Fundamental Algorithms

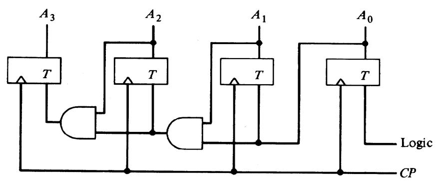
For Example



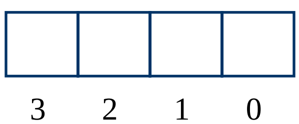
S - 10

Ripple Counter





Software Ripple Counter*

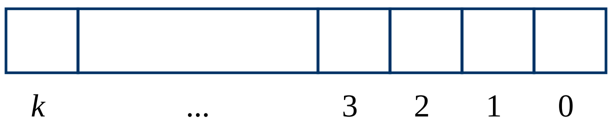



```
Increment(A)
    i ← 0
    while i<length[A] and A[i]=1
        do A[i] ← 0
           i ← i+1
    if i < length[A]
        then A[i] ← 1
```

*Cost of an increment is linear in the number of flipped bits.
Total number of flips for a sequence of *n* increments?

Amortized Cost

- We seek a potential function Φ so that $\Phi(D_i) \geq \Phi(D_0) \geq 0$.



Fundamental Algorithms

Total Amortized Cost

- Suppose the i^{th} `Increment` operation resets t_i bits, so $c_i \leq t_i + 1$. The number of one's in counter after i^{th} iteration is $b_i \leq b_{i-1} - t_i + 1$.

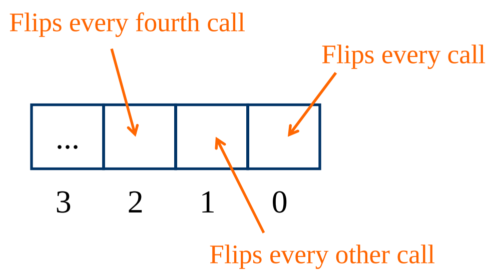
- The potential difference is

$$\begin{aligned}\Phi(D_i) - \Phi(D_{i-1}) &= b_i - b_{i-1} \\ &\leq (b_{i-1} - t_i + 1) - b_{i-1} \\ &= 1 - t_i\end{aligned}$$

- The amortized cost is therefore

$$\begin{aligned}\hat{c}_i &= c_i + \Phi(D_i) - \Phi(D_{i-1}) \\ &\leq (t_i + 1) + (1 - t_i) \\ &= 2\end{aligned}$$

Aggregate Method



S - 15