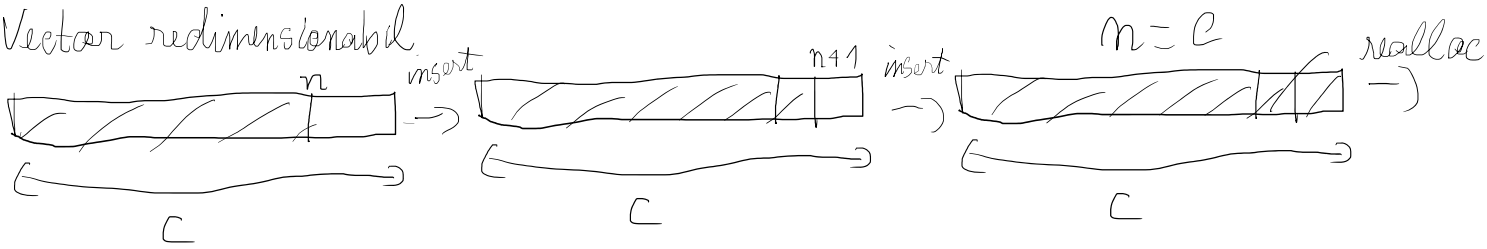
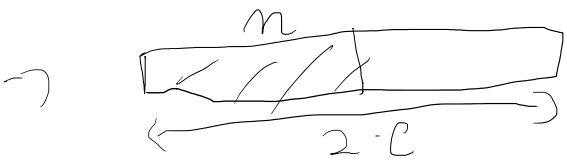
De ce analiza amortizata?

Friday, 19 November 2021 10.00

Vector redimensionabil





2 operatii [push O(1)
resize O(n)

k operatii → O(k · n)

ca să fac resize trb. să fac n op. push

$1 + 1 + 1 + \ldots + 1 + n = 2n = O(n)$

($1 + 1 + 1 + \ldots + 1$: n)

Stiva cu mpop

Friday, 19 November 2021 10.01

stivă

push(x) -> adaugă pe stivă -> O(1)

pop() -> O(1)

mpop(k) = k · pop() = scot k elemente de pe stivă (min(k, n))

```
mpop(k) {
    for(int i=0; i<min(k,n); ++i){
        S.pop();
    }
} -> O(min(k,n))
```

n operații => worst case n mpop => $O(n^2)$?

cu op. de mpop nu pot să scot mai multe el. decât sunt pe stivă (am dat push înainte)

pt. n operatii => worst-case $T(n)$

cost amortizat $\hat{c} = \frac{T(n)}{n}$

obs: cost amortizat identic pt toate op.

ex: stiva cu mpop

n operatii push și mpop

$T(n) = T(push) + T(mpop)$

dar $T(mpop) \leq T(push)$ (nu pot să scot mai multe el. decât sunt pe stivă)

$T(n) \leq 2 \cdot T(push) = 2 \cdot n \cdot 1$

$\hat{c} = \frac{T(n)}{n} = \frac{2n}{n} = 2 \in O(1)$

4.1 O problemă oarecare

Să presupunem că avem de efectuat o succesiune de n operații pe o structură de date în care operația cu numărul i are asociat costul c_i de mai jos.

$$c_i = \begin{cases} i, & \text{dacă } i \text{ este putere a lui 2} \\ 1, & \text{altfel} \end{cases}$$

Folosiți metoda agregării și/sau metoda creditelor pentru a determina costul amortizat per operație.

a) n operatii

$\log n$ op. cu $c_i = i$

$n - \log n$ op. cu $c_i = 1$

$\hat{c} = \frac{T(n)}{n} = \frac{3n - \log n - 1}{n} \in O(1)$

$T(n) = T(c_i = i) + T(c_i = 1) =$

$= \sum_{i=0}^{\log n} 2^i + (n - \log n) \cdot 1 =$

$= 2^{\log n + 1} - 1 + n - \log n =$

$= 2 \cdot 2^{\log n} - 1 + n - \log n =$

$= 2n - 1 + n - \log n =$

$= 3n - \log n - 1 = O(n)$

b) dar dacă $c_i = i$ dacă i e multiplu de k

$\frac{n}{k}$ op. cu $c_i = i$

$n - \frac{n}{k}$ op. cu $c_i = 1$

$\hat{c} = \frac{T(n)}{n} = O(1)$

$T(n) = T(c_i = 1) + T(c_i = i) =$

$= n - \frac{n}{k} + \sum_{i=0}^{\frac{n}{k}} k \cdot i =$

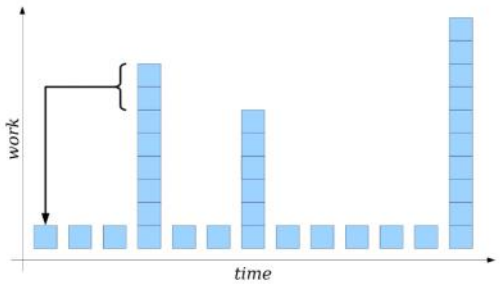
$= n - \frac{n}{k} + k \sum_{i=0}^{\frac{n}{k}} i = n - \frac{n}{k} + k \cdot \frac{\frac{n}{k} \cdot \left(\frac{n}{k} + 1\right)}{2}$

$= n - \frac{n}{k} + \frac{n \cdot (n+k)}{2k} =$

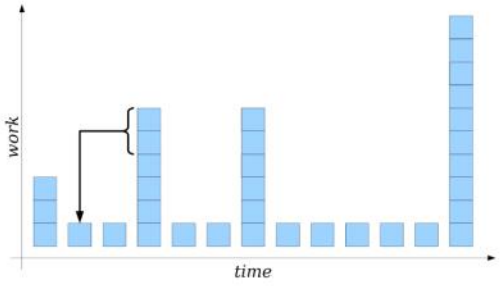
$\hat{C} = \frac{T(n)}{n} = O(n)$

$= n - \frac{n}{k} + \frac{n \cdot (n+1)}{2k} =$

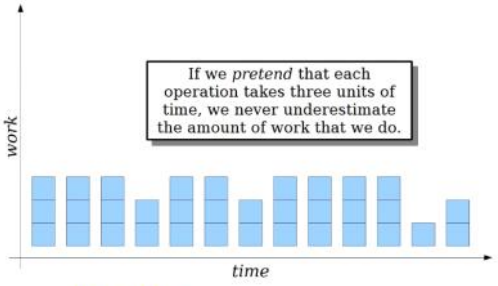
$= n - \frac{n}{k} + \frac{n^2}{2k} + \frac{nk}{2k} \in O(n^2)$



Key Idea: Backcharge expensive operations to cheaper ones.



Key Idea: Backcharge expensive operations to cheaper ones.



Key Idea: Backcharge expensive operations to cheaper ones.

idee: pot să plătesc op. mai mari anticipat

credit = $\hat{c}_i - c_i$

obs: cost amortizat pt. fiecare op.

c_i = costul real

presupunem că știm $\hat{c}_i$

$\hat{c}_i > c_i \Rightarrow$ credit $> 0 \Rightarrow$ depun credit

$\hat{c}_i < c_i \Rightarrow$ credit $< 0 \Rightarrow$ retrag credit (consum)

credit ≥ 0 la orice pas (nu rămân fără credit)

$\sum \hat{c}_i \geq \sum c_i$

ex: Stiva cu mpop

$c_{push} = 1$ $\quad \hat{c}_{push} = 2$

$c_{mpop} = n$ $\quad \hat{c}_{mpop} = 0$

$credit_{push} = \hat{c}_{push} - c_{push} = 1$

$credit_{mpop} = \hat{c}_{mpop} - c_{mpop} = -n$

pot să rămân cu creditul pe $<?$ (< 0)

n op. push $\Rightarrow$ credit $= n$ $\quad n - n = 0 \geq 0$

mpop $\Rightarrow -n$

Nu, tot timpul credit ≥ 0

(nu pot să dau mai multe pop decât sunt el. pe stivă)

Metoda potentialului

Friday, 19 November 2021 11.06

duală metodei creditelor

analogie: hidrocentrală (consum energie pt a pompa apa în sus și a stoca sub formă de energie potențială pt a o consuma ulterior

asociez structurii de date o fn. potențial $\Phi(S)$

$\hat{c}_i = c_i + \underbrace{\Phi(S_i) - \Phi(S_{i-1})}_{\text{dif de potențial}}$

S_0 = potențial inițial (de obicei 0)

condiție: să nu cad sub pragul inițial de potențial

$\hat{c}_1 = c_1 + \Phi(S_1) - \Phi(S_0)$

$\hat{c}_2 = c_2 + \Phi(S_2) - \Phi(S_1)$

$\sum \hat{c}_i = \sum c_i + \Phi(S_n) - \Phi(S_0)$

$\Phi(S_n) \geq \Phi(S_0)$

de obicei iau $\Phi(S_0) = 0$

$\Rightarrow \Phi(S_n) \geq 0$

→ ales. fn. de potențial

$\Phi(S)$ = nr. de elemente din stivă

push:

$\hat{c}_{push} = c_{push} + \Phi(S_i) - \Phi(S_{i-1})$

presupun că am n el. în stivă ($\Phi(S_{i-1}) = n$) $\Rightarrow$ după push $\Rightarrow \Phi(S_i) = n+1$

$\hat{c}_{push} = 1 + n + 1 - n = 2$

mpop:

$\hat{c}_{mpop} = c_{mpop} + \Phi(S_i) - \Phi(S_{i-1})$

$\Phi(S_{i-1}) = n$ (presupun)

$\Phi(S_i) = n - k$ sau 0

$= n - \min(n,k)$

$\Phi(S_i) - \Phi(S_{i-1}) = n - \min(n,k) - n$

$= -\min(n,k)$

$\hat{c}_{mpop} = c_{mpop} - \min(n,k) =$

$= \min(n,k) - \min(n,k)$

$= 0$

$\Phi(S_0) = 0$

$\Phi(S_i) \geq 0 \ \forall i$

nu pot să scot mai multe el. decât sunt pe stivă

Exercitii

Friday, 19 November 2021 11.25

Contor binar pe k biti

```
function Inc(A) {
	// i = rangul binar curent
	i = 0;

	// daca adun 1, trebuie sa propag transportul
	// pana cand gasesc un zero
	// 0 + 1 = 1 si nu mai am transport
	while (i < A.length && A[i] == 1) {
		A[i] = 0;
		i++;
	}

	// am gasit un zero daca nu am facut overflow
	if (i < A.length) {
		A[i] = 1;
	}
}
```

0 0 0 0 0
0 0 0 0 1
0 0 0 1 0
0 0 0 1 1
0 0 1 0 0
0 0 1 0 1
0 0 1 1 0
0 0 1 1 1

cost pt. 1 inc $\Rightarrow O(n)$

n inc $\Rightarrow O(n^2)$?

0 1 1 1 1 ... 1 1 + 1
(n)
1 0 0 0 ... 0
(n)

Analiza agregată

$$T(n) = \frac{n}{2^0} + \frac{n}{2^1} + \frac{n}{2^2} + \dots + \frac{n}{2^k} = \sum_{i=0}^{k} \frac{n}{2^i} = n \cdot \sum_{i=0}^{k} \frac{1}{2^i} \leq n \cdot \sum_{i=0}^{\infty} \frac{1}{2^i} = 2n$$

$\hat{c} = \frac{T(n)}{n} = 2 \in O(1)$

Metoda creditelor

2 op: set, reset

nu pot să fac reset la mai multi biti decât am făcut set

$c_{set} = 1$ $\quad \hat{c}_{set} = 2$

$c_{reset} = n$ $\quad \hat{c}_{reset} = 0$

Metoda potențialului

$\Phi(S_i)$ = nr. de biți de 1

Inițial am b_{i-1} biți de 1 în contor

În pasul curent, resetez t_i biți

$c_{inc} = t_i + 1$

$\Phi(S_{i-1}) = b_{i-1}$

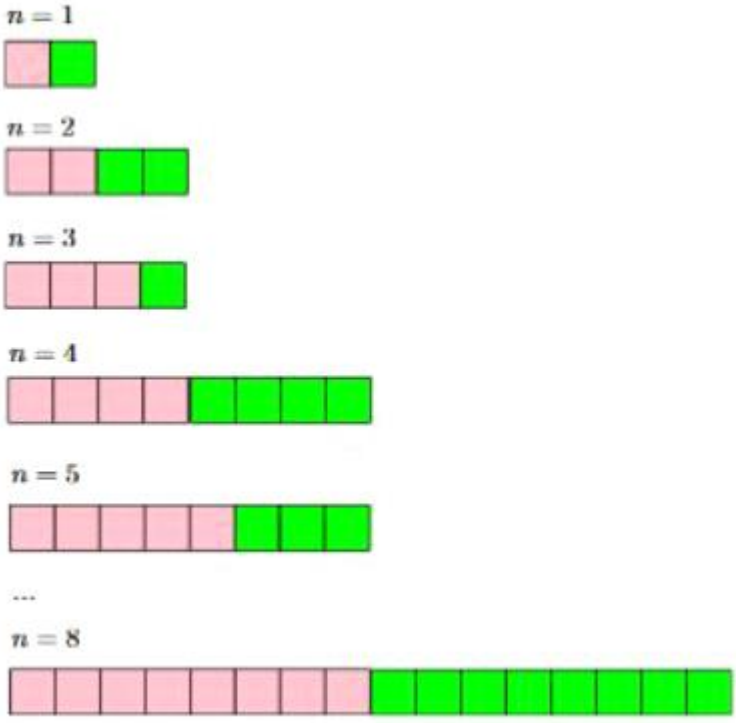
$\Phi(S_i) = b_{i-1} - t_i + 1$

$\hat{c}_{mc} = c_{mc} + \phi(s_i) - \phi(s_{i-1}) = t_i + 1 + \Phi_{i-1} - \Phi_i - \Phi_{i-1}$

$= 2 \in O(1)$

Vector redimensionabil

Friday, 19 November 2021 11.42



Alt exercitiu

Friday, 19 November 2021 11.47

1. [3p] Orasul Chicago are multe cladiri, dar numai unele dintre ele au **vedere buna** catre lacul Michigan. Sa presupunem ca avem un vector A[1..n] care stocheaza inaltimile celor n cladiri din oras (acestea sunt indexate de la vest la est).

Cladirea cu numarul i are o vedere buna catre lacul Michigan **daca si numai daca** fiecare cladire de la estul ei este mai scunda. Se considera algoritmul prezentat in pseudocodul de mai jos care calculeaza care cladire are vedere buna catre lacul Michigan.

```
GoodView(A[1..n]) {
        S = stiva goala
        for (i = 1; i <= n; ++i) {
                insert(S, A, i); // insereaza in stiva elementul A[i]
        }
        // numerele ramase in stiva reprezina indicii cladirilor cautate
        return S.getAllElements();
}
Insert(S, A, i) {
        while (S.empty() == false && A[i] > A[ S.top() ] {
                S.pop();
        }
        S.push(i);
}
```

Care este costul amortizat al operatiei Insert? Care este costul total al secventei de n operatii Insert? (al functiei GoodView).