3. The potential method

Potential function ϕ

$\phi : D \longrightarrow \mathbb{R}^+$ we allow $\phi_0 \le 0$

Data structure D → $\phi(D)$

Idea: Potential ~ Credit.

t_i = actual cost of the i-th operation

ϕ_i = potential after execution of the i-th operation (= $\phi(D_i)$)

$\phi_i = \phi(D_i)$

a_i = amortized cost of the i-th operation

$$\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} (t_i + \phi_i - \phi_{i-1})$$

telescoping series

$$= \sum_{i=1}^{n} t_i + \underbrace{\phi_n - \phi_0}_{\geq 0}$$

$$\Rightarrow \sum_{i=1}^{n} t_i \le \sum_{i=1}^{n} a_i .$$

Definition:

actual cost

$$a_i = t_i + \underbrace{\phi_i - \phi_{i-1}}_{\text{change of the potential function}}$$

Example: binary counter

D_i = counter value after the *i*-th operation
$\phi_i = \phi(D_i)$ = # of 1's in D_i

i–th operation	# of 1's
D_{i-1}: .....0/1.....01.....1	B_{i-1}
D_i : .....0/1.....10.....0	$B_i = B_{i-1} - b_i + 1$

t_i = **actual bit flip cost of operation *i***
= b_i+1

Binary counter

t_i = actual bit flip cost of operation *i*
a_i = amortized bit flip cost of operation *i*

$$
\begin{aligned}
a_i &= (b_i + 1) + (B_{i-1} - b_i + 1) - B_{i-1} \\
&= 2 \\
&\Rightarrow \sum t_i \le 2n
\end{aligned}
$$

Dynamic tables

Problem:

Maintain a table supporting the operations insert and delete such that

- the table size can be adjusted dynamically to the number of items
- the used space in the table is always at least a constant fraction of the total space
- the total cost of a sequence of *n* operations (insert or delete) is *O(n)*.

$O(1)$ average cost per operation

Table should be able to grow and shrink

Applications: hash table, heap, stack, etc.

Load factor α_T: number of items stored in the table divided by the size of the table

Idea When table is full and an insert occurs → double table size.
Involves copying ~~...~~ the entire old table to the new table.

Implementation of 'insert'

```
class dynamic table {

    int [] table;

    int size;                    // size of the table
    int num;                     // number of items

dynamicTable() {                 // initialization of an empty table
    table = new int [1];
    size = 1;
    num = 0;
    }
```

Implementation of ‘insert’

```
insert ( int x) {
    if (num == size ) {                         // table is full
        newTable = new int [2*size];
        for (i = 0; i < size; i++)              // Copy the elements of table into
                                                //   new Table
            newTable[i] = table[i];             // O(num) = O(size)
        table = newTable;
        size = 2*size;
    }
    table[num] = x;                             // O(1) time for insert,
    num = num + 1;
}
```

Cost of n insertions into initially empty table

t_i = cost of the i-th insert operation

Worst case:

insert

$t_i = 1$ insert if the table is not full prior to operation i

$t_i = (i - 1) + 1 = i$ if the table is full prior to operation i.

copying from old to new table

Thus n insertions incur a total cost of at most

$$\sum_{i=1}^{n} i = \Theta(n^2)$$ $\Theta(n)$ per operation overestimate!

Amortized worst case:

Aggregate method, accounting method, potential method

Potential method

Goal: Want to show amortized cost $a_i = O(1)$

T table with

- $k = T.num$ items
- $s = T.size$ size

Potential function

$$\phi(T) = 2 \cdot k - s$$

Here the potential function "falls from the sky"! We take that for granted.

Finding good potential functions is "an art"!

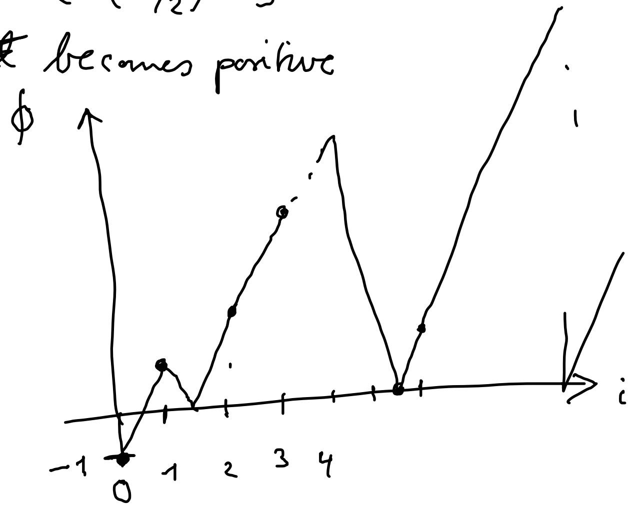
Potential method

$\boxed{\phi = 2 \cdot k - s}$

Properties

- $\phi_0 = \phi(T_0) = \phi$ (empty table) = -1
- Immediately before a table expansion we have $k = s$,
  thus $\phi(T) = k = s.$
  $\phi = 2 \cdot k - s = 2 \cdot s - s = s$
- Immediately after a table expansion we have $k = s/2$,
  thus $\phi(T) = 2k - s = 0$.
  $\phi = 2 \cdot k - s = 2 \cdot (s/2) - s = 0$
  After that we insert an element and ϕ ~~tat~~ becomes positive
- For all $i \geq 1$: $\phi_i = \phi(T_i) > 0$
  Since $\phi_n - \phi_0 \geq 0$,

$$\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} (t_i + \phi_i - \phi_{i-1}) = \sum_{i=1}^{n} t_i + \underbrace{\phi_n - \phi_0}_{\geq 0} \Rightarrow \sum_{i=1}^{n} t_i \leq \sum_{i=1}^{n} a_i$$



Amortized cost a_i of the *i*-th insertion

Will show: $a_i \leq 3 \quad \Rightarrow \quad \sum_{i=1}^{n} t_i \leq \sum_{i=1}^{n} a_i \leq 3 \cdot n \quad \Rightarrow \frac{1}{n} \sum_{i=1}^{n} t_i \leq 3.$

k_i = # items stored in T after the *i*-th operation

s_i = table size of T after the *i*-th operation

Case 1: *i*-th operation does not trigger an expansion

$k_i = k_{i-1} + 1, \; s_i = s_{i-1}$

$\phi = 2 \cdot k - s$

$a_i = t_i + \phi_i - \phi_{i-1}$

insert $\quad \phi_i \quad \phi_{i-1}$

$s_i = s_{i-1}$

$a_i = 1 + (2k_i - \cancel{s_i}) - (2k_{i-1} - \cancel{s_{i-1}})$

$= 1 + 2(k_i - k_{i-1}) = 1 + 2 \cdot (k_{i-1} + 1 - k_{i-1}) = 1 + 2$

$= 3$

Case 2: *i*-th operation does trigger an expansion

$k_i = k_{i-1} + 1,\ s_i = 2s_{i-1}$, $k_{i-1} = s_{i-1}$

$\Phi = 2 \cdot k - s$

$a_i = t_i + \Phi_i - \Phi_{i-1}$

$$a_i = \underbrace{k_{i-1} + 1}^{\text{copying + insert}} + \overbrace{(2k_i - s_i)}^{\phi_i} - \overbrace{(2k_{i-1} - s_{i-1})}^{\phi_{i-1}}$$

$= k_{i-1} + 1 + (2(k_{i-1} + 1) - 2s_{i-1}) - (2k_{i-1} - s_{i-1})$

$= k_{i-1} \cdot (1 + 2 - 2) + s_{i-1} \cdot (-2 - (-1)) + 3$

$= k_{i-1} - s_{i-1} + 3$

$= 3$ ∎

Inserting and deleting items

Now: Contract the table whenever the load becomes too small.

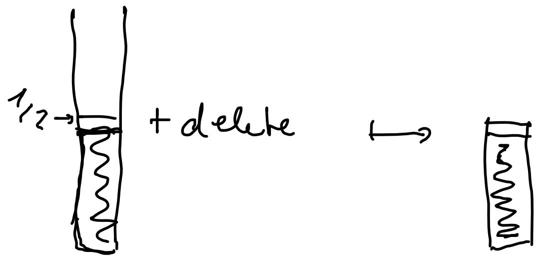
Goal:

(1) The load factor is bounded from below by a constant. e.g. $1/2$, $1/4$, ...

(2) The amortized cost of a table operation is constant.

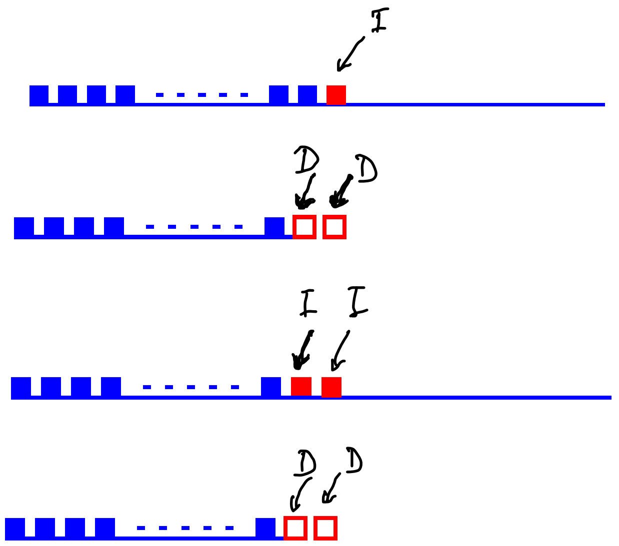
First approach

- Expansion: as before
- Contraction: Halve the table size when a deletion would cause the table to become less than half full.



„Bad“ sequence of table operations

Sequence of $n+1$ operations



	Cost
n/2 ‘insert’ op. (table is full)	3 *n/2*
I: expansion	*n/2* + 1
D, D: contraction	*n/2* + 1
I, I : expansion	*n/2* + 1
D, D: contraction	

$\frac{n}{4}$

Total cost of the sequence of operations:
$I_{n/2}, I, D, D, I, I, D, D, \ldots$ of length n is

$$\geq \frac{n}{4} \cdot \frac{n}{2} = \Theta(n^2)$$

$\Rightarrow$ average cost $\Theta(n)$

Second approach

Expansion: Double the table size when an item is inserted into a full table.

Contraction: Halve the table size when a deletion causes the table to become less than ¼ full.

Property: At any time the table is at least ¼ full, i.e.

$$\frac{1}{4} \leq \alpha(T) \leq 1$$

What is the cost of a sequence of table operations?

Goal: Show constant amortized cost.

Analysis of ‘insert’ and ‘delete’ operations

\# elements

k = *T.num*, *s* = *T.size*, $\alpha = k/s$

load factor size

Potential function ϕ

$$\boxed{\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}}$$

Is $\phi \geq 0$ always?

Case 1: $\alpha \geq 1/2 \Rightarrow k \geq s/2 \Rightarrow 2 \cdot k - s \geq 0$ ✓

Case 2: $\alpha < 1/2 \Rightarrow k < s/2 \Rightarrow s/2 - k \geq 0$ ✓

Analysis of ‘insert’ and ‘delete’ operations

$$\boxed{\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}}$$

Immediately after a table expansion or contraction:

$s = 2k$, thus $\phi(T) = 0$

$s = 2k \Rightarrow \alpha = \frac{k}{s} = \frac{1}{2}$

$2 \cdot k - s = 2 \cdot k - 2k = 0$

Analysis of an ‘insert’ operation

Want to show: $a_i \leq 3$

i-th operation: $k_i = k_{i-1} + 1$

Case 1: $\alpha_{i-1} \geq ½$ Have done the analysis already ✓

Case 2: $\alpha_{i-1} < ½$

Case 2.1: $\alpha_i < ½$

Case 2.2: $\alpha_i \geq ½$

Analysis of an 'insert' operation

Case 2.1: $\alpha_{i-1} < ½$, $\alpha_i < ½$ no expansion

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}$$

$\leftarrow \alpha_{i-1}, \alpha_i$

$$a_i = 1 + \left(\frac{s_i}{2} - k_i\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right)$$

$$= 1 + \left(\frac{s_{i-1}}{2} - (k_{i-1}+1)\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right)$$

$$= 1 + (-1)$$

$$= 0$$

$$\leq 3 \checkmark$$

$k_i = k_{i-1} + 1$

$s_i = s_{i-1}$

Analysis of an 'insert' operation

Case 2.2: $\alpha_{i-1} < ½$, $\alpha_i \geq ½$ no expansion

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 & \leftarrow \alpha_i \\ s/2 - k, \text{if } \alpha < 1/2 & \leftarrow \alpha_{i-1} \end{cases}$$

$$a_i = 1 + (2 \cdot k_i - s_i) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right)$$

$$= 1 + (2 \cdot (k_{i-1} + 1) - s_{i-1}) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right)$$

$$= 3 + \underbrace{3 \cdot k_{i-1} - \frac{3}{2} \cdot s_{i-1}}_{\leq 0}$$

$$\leq 3 \checkmark$$

$s_i = s_{i-1}$

$k_i = k_{i-1} + 1$

$\alpha_{i-1} < \frac{1}{2}$

$\Rightarrow k_{i-1} < \frac{s_{i-1}}{2}$

Analysis of a ‘delete’ operation

$k_i = k_{i-1}$ - 1

Case 1: $\alpha_{i-1} < ½$,

Case 1.1: deletion does not trigger a contraction

$s_i = s_{i-1}$ $\alpha_i < \frac{1}{2}$

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}$$

$\leftarrow \alpha_{i-1}, \alpha_i$

$$\begin{aligned} a_i &= 1 + \left(\frac{s_i}{2} - k_i\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right) \\ &= 1 + \left(\frac{s_{i-1}}{2} - (k_{i-1} - 1)\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right) \\ &= 1 + (-(-1)) \\ &= 2 \\ &\leq 3 \end{aligned}$$

$s_i = s_{i-1}$

$k_i = k_{i-1} - 1$

Analysis of a 'delete' operation

$k_i = k_{i-1} - 1$

Case 1: $\alpha_{i-1} < ½$

Case 1.2: $\alpha_{i-1} < ½$ deletion does trigger a contraction

$s_i = s_{i-1}/2 \quad k_{i-1} = s_{i-1}/4$

$$\alpha_i = \frac{k_i}{s_i} = \frac{k_{i-1}-1}{s_{i-1}/2} = \frac{s_{i-1}/4 - 1}{s_{i-1}/2} < \frac{1}{2}$$

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}$$

$\leftarrow \alpha_{i-1}, \alpha_i$

$$\begin{aligned} a_i &= 1 + k_{i-1} + \left(\frac{s_i}{2} - k_i\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right) \\ &= 1 + k_{i-1} + \left(\frac{s_{i-1}}{4} - (k_{i-1} - 1)\right) - \left(\frac{s_{i-1}}{2} - k_{i-1}\right) \\ &= 2 + k_{i-1} - \frac{s_{i-1}}{4} \\ &= 2 \leq 3 \checkmark \end{aligned}$$

Analysis of a 'delete' operation

Case 2: $\alpha_{i-1} \geq ½$ no contraction

$$s_i = s_{i-1},\ k_i = k_{i-1} - 1$$

Case 2.1: $\alpha_i \geq ½$

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 \\ s/2 - k, \text{if } \alpha < 1/2 \end{cases}$$

$\leftarrow \alpha_{i-1}, \alpha_i$

$$\begin{aligned} a_i &= 1 + (2k_i - s_i) - (2k_{i-1} - s_{i-1}) \\ &= 1 + (2 \cdot (k_{i-1} - 1) - s_{i-1}) - (2k_{i-1} - s_{i-1}) \\ &= 1 - 2 = -1 \\ &\leq 3 \quad ✓ \end{aligned}$$

$s_i = s_{i-1}$

$k_i = k_{i-1} - 1$

Analysis of a ‘delete’ operation

Case 2: $\alpha_{i-1} \geq ½$ no contraction

$s_i = s_{i-1} \quad k_i = k_{i-1} - 1$

Case 2.2: $\alpha_i < ½$

Potential function ϕ

$$\phi(T) = \begin{cases} 2k - s, \text{if } \alpha \geq 1/2 & \leftarrow \alpha_{i-1} \\ s/2 - k, \text{if } \alpha < 1/2 & \leftarrow \alpha_i \end{cases}$$

$$a_i = 1 + \left(\frac{s_i}{2} - k_i\right) - \left(2k_{i-1} - s_{i-1}\right)$$

$$= 1 + \left(\frac{s_{i-1}}{2} - (k_{i-1} - 1)\right) - \left(2 \cdot k_{i-1} - s_{i-1}\right)$$

$$= 2 + \underbrace{\frac{3}{2} \cdot s_{i-1} - 3 \cdot k_{i-1}}_{\leq 0}$$

$$\leq 2 \leq 3 \checkmark$$

$s_i = s_{i-1}$

$k_i = k_{i-1} - 1$

$\alpha_{i-1} \geq 1/2$

$\frac{k_{i-1}}{s_{i-1}} \geq 1/2$