1. análisis agregado
2. método de contabilidad
3. método potencial

- STACK (PILA)
    - Push(S,x)
    - Pop(S) -> x
    - Multipop(S,k)

- BINARY COUNTER (CONTADOR BINARIO)
    - Increment(A)

- DYNAMIC TABLE (TABLA DINÁMICA)
    - Table-insert(T,ítem)
    - Table-delete(T,ítem)

Análisis amortizado

Dada una estructura de datos S y un conjunto de operaciones sobre S. Ahora dada una secuencia de operaciones realizadas, se determina el costo promedio por operación. Sea el costo total de las n operaciones de la secuencia está acotado superiormente por T(N) entonces el costo promedio es T(n)/n.

- STACK (PILA)
  - Push(S,x) 1
  - Pop(S) -> x 1
  - Multipop(S,k) min(|S|,k)

¿$T(n)=O(n^2)$ o $O(n)$?

```
MULTIPOP(S, k)
while not STACK-EMPTY(S) and k > 0
    POP(S)
    k = k - 1
```

Análisis amortizado

- BINARY COUNTER (CONTADOR BINARIO)
  - Increment(A)

A es un arreglo de K bits binarios, A[0] es el dígito menos significativo y A[k-1] el más significativo. A.length=k.

```
INCREMENT(A)
i = 0
while i < A.length and A[i] == 1
    A[i] = 0
    i = i + 1
if i < A.length
    A[i] = 1
```

Counter value	A[7]	A[6]	A[5]	A[4]	A[3]	A[2]	A[1]	A[0]	Total cost
0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	1	1
2	0	0	0	0	0	0	1	0	3
3	0	0	0	0	0	0	1	1	4
4	0	0	0	0	0	1	0	0	7
5	0	0	0	0	0	1	0	1	8
6	0	0	0	0	0	1	1	0	10
7	0	0	0	0	0	1	1	1	11
8	0	0	0	0	1	0	0	0	15
9	0	0	0	0	1	0	0	1	16
10	0	0	0	0	1	0	1	0	18
11	0	0	0	0	1	0	1	1	19
12	0	0	0	0	1	1	0	0	22
13	0	0	0	0	1	1	0	1	23
14	0	0	0	0	1	1	1	0	25
15	0	0	0	0	1	1	1	1	26
16	0	0	0	1	0	0	0	0	31

¿T(n)=O(kn) o O(n)?

Método de contabilidad

Se define una carga ĉ diferente para cada operación que puede distinta al costo real C de la operación.

$$\sum_{i=1}^{n} \hat{c}_i \geq \sum_{i=1}^{n} c_i$$

- STACK (PILA)
  - Push(S,x) 1 2
  - Pop(S) -> x 1 0
  - Multipop(S,k) min(|S|,k) 0

- BINARY COUNTER (CONTADOR BINARIO)
  - Increment(A) 2

Método potencial

Dada una estructura de datos D, definir una función de “energía potencial” ϕ sobre D. Sea D_0 el estado inicial de D, y Di el estado de D después de aplicar i-ésima operación. Entonces el costo amortizado de la operación i-ésima es:

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

Hay que definir adecuadamente tal que $\phi(D_i) \geq 0$ para cualquier i.

Para Stack $\phi(S)=|S|$ entonces $\phi(S_o)=0$

$\phi(S_i) - \phi(S_{i-1}) = (|S_{i-1}| + 1) - |S_{i-1}| = 1$ para Push(S_{i-1},x)

$\phi(S_i) - \phi(S_{i-1}) = -1$ para Pop(S_{i-1})

$\phi(S_i) - \phi(S_{i-1}) = \min(|S_{i-1}|, k)$ para Multipop(S_{i-1},k)

Para Binary counter $\phi(A)$=cantidad de bit 1 en A entonces $\phi(A_o)=0$

$\phi(A_i) - \phi(A_{i-1}) \leq (\phi(A_{i-1}) - t_i + 1) - \phi(A_{i-1}) = 1 - t_i$ siendo t_i la cantidad de iteraciones dentro loop while de la Increment para la i-ésima operación.

Dynamic Table (solamente con Table-insert)

α Factor de carga T.num/T.size : T.size: tamaño del slot actual, T.num: número de elementos

- Solamente Table-Inserts

1. análisis agregado

   T(n)=3n

2. método de contabilidad

   ĉ = 3

3. método potencial

$$\Phi(T) = 2 \cdot T.num - T.size$$

$$\begin{aligned}
\hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\
&= 1 + (2 \cdot num_i - size_i) - (2 \cdot num_{i-1} - size_{i-1}) \\
&= 1 + (2 \cdot num_i - size_i) - (2(num_i - 1) - size_i) \\
&= 3 .
\end{aligned}$$

$$\begin{aligned}
\hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\
&= num_i + (2 \cdot num_i - size_i) - (2 \cdot num_{i-1} - size_{i-1}) \\
&= num_i + (2 \cdot num_i - 2 \cdot (num_i - 1)) - (2(num_i - 1) - (num_i - 1)) \\
&= num_i + 2 - (num_i - 1) \\
&= 3 .
\end{aligned}$$

Dynamic Table (con Table-insert y Table-delete)

- ***α*** debe estar acotado inferiormente por un valor constante > 0
- ***ĉ*** debe estar acotado superiormente por un valor constante

$$\Phi(T) = \begin{cases} 2 \cdot T.num - T.size & \text{if } \alpha(T) \geq 1/2 \,, \\ T.size/2 - T.num & \text{if } \alpha(T) < 1/2 \,. \end{cases}$$

Observaciones:

- cuando $\alpha(T) = ½$, entonces $T.size = 2T.num$ y $\phi(T) = 0$
- cuando $\alpha(T) = 1$, entonces $T.size = T.num$, $\phi(T) = T.num$ y podría pagar una expansión
- cuando $\alpha(T) = ¼$, entonces $T.size = 4T.num$, $\phi(T) = T.num$ y podría pagar una contracción

- Si i-ésimo operación es Table-insert y $\alpha(T_{i-1}) \geq ½$ entonces $\hat{c}_i \leq 3$ independientemente si se expande o no.
  - Si $\alpha(T_i) < ½$

$$\begin{aligned} \hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\ &= 1 + (size_i/2 - num_i) - (size_{i-1}/2 - num_{i-1}) \\ &= 1 + (size_i/2 - num_i) - (size_i/2 - (num_i - 1)) \\ &= 0 \,. \end{aligned}$$

Dynamic Table (con Table-insert y Table-delete)

- Si α(Ti-1) < ½ y α(Ti) ≥ ½

$$
\begin{aligned}
\hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\
&= 1 + (2 \cdot num_i - size_i) - (size_{i-1}/2 - num_{i-1}) \\
&= 1 + (2(num_{i-1} + 1) - size_{i-1}) - (size_{i-1}/2 - num_{i-1}) \\
&= 3 \cdot num_{i-1} - \frac{3}{2} size_{i-1} + 3 \\
&= 3\alpha_{i-1} size_{i-1} - \frac{3}{2} size_{i-1} + 3 \\
&< \frac{3}{2} size_{i-1} - \frac{3}{2} size_{i-1} + 3 \\
&= 3 .
\end{aligned}
$$

En todos los casos $\hat{c}_i \leq 3$

- Si i-ésimo operación es Table-delete
  - Si α(Ti-1) < ½ y α(Ti) ≥ ¼

$$
\begin{aligned}
\hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\
&= 1 + (size_i/2 - num_i) - (size_{i-1}/2 - num_{i-1}) \\
&= 1 + (size_i/2 - num_i) - (size_i/2 - (num_i + 1)) \\
&= 2 .
\end{aligned}
$$

Dynamic Table (con Table-insert y Table-delete)

- Si α(Ti-1) = ¼

$$
\begin{aligned}
\hat{c}_i &= c_i + \Phi_i - \Phi_{i-1} \\
&= (num_i + 1) + (size_i/2 - num_i) - (size_{i-1}/2 - num_{i-1}) \\
&= (num_i + 1) + ((num_i + 1) - num_i) - ((2 \cdot num_i + 2) - (num_i + 1)) \\
&= 1 .
\end{aligned}
$$

- Si α(Ti-1) ≥ ½ (ejercicio 17.4-2) . Hay que considerar 2 subcasos (I) α(Ti) ≥ ½ y (II) α(Ti) < ½