Análisis amortizado: Contador Binario

Teoría de Algoritmos I (75.29 / 95.06)

Ing. Víctor Daniel Podberezski
vpodberezski@fi.uba.ar

Contador binario

Tenemos un contador binario

Utilizamos un vector A de k bits

6	5	4	3	2	1	0

Comienza en cero

Para incrementar en 1:

```
i = 0
Mientras i < k y A[i]=1
    A[i] = 0;
    i++
if i < k
    A[i] = 1;
```

En el peor caso la complejidad del contador es O(k).

Realizar n operaciones es O(nk) **NO!!**

Contador binario - Aggregate analysis

Contador	A[3]	A[2]	A[1]	A[0]	Costo operación	Costo acumulado
0	0	0	0	0	0	0
1	0	0	0	1	1	1
2	0	0	1	0	2	3
3	0	0	1	1	1	4
4	0	1	0	0	3	7
5	0	1	0	1	1	8
6	0	1	1	0	2	10
7	0	1	1	1	1	11
8	1	0	0	0	4	15
9	1	0	0	1	1	16
10	1	0	1	0	2	18

“n/8”

“n/4”

Cambia siempre (“n” veces)

Cambia la mitad de las veces (“n/2”)

Contador binario - Aggregate analysis (cont.)

En general, el bit A[i] cambio $\lfloor n/2^i \rfloor$ veces

- En una secuencia de n operaciones de incremento

La cantidad total de cambios de bits en n operaciones

- En una secuencia de n operaciones de incremento

$$\sum_{i=0}^{k-1} \left\lfloor \frac{n}{2^i} \right\rfloor < n * \sum_{i=0}^{\infty} \left(\frac{1}{2^i} \right) = 2n$$

Realizar n operaciones es – por lo tanto – O(n)

- El costo amortizado de cada operación es O(1)

Contador binario - Accounting method

Los costos de la operación “incrementar” son variables

- Dependen de la cantidad de bits modificados

Proponemos:

- Costo amortizado de 2 por cada bit cambiado a 1.
- Inicialmente el crédito es 0
- No hay operaciones con crédito negativo (en el caso extremo no hay bits a 1).
- Los costos de cambiar a 0 se “pagan” con lo ahorrado en el cambio a bits a 1.

Contador binario - Accounting method (cont.)

Contador	A[3]	A[2]	A[1]	A[0]	Costo operación	Costo amortizado	Crédito
0	0	0	0	0	0	0	0
1	0	0	0	1	1	2	1
2	0	0	1	0	2	2	1
3	0	0	1	1	1	2	2
4	0	1	0	0	3	2	1
5	0	1	0	1	1	2	2
6	0	1	1	0	2	2	2
7	0	1	1	1	1	2	3
8	1	0	0	0	4	2	1
9	1	0	0	1	1	2	2
10	1	0	1	0	2	2	2

Contador binario - Accounting method (cont)

Se cumple – como se requiere - que $\sum_{i=1}^{n} \hat{C}_i \geqslant \sum_{i=1}^{n} C_i$

Ademas tenemos que $T(n) = \sum_{i=1}^{n} \hat{C}_i$ es O(n)

Y T(n)/n = O(1)

Contador binario - Potential method

Definimos:

$\Phi(D_i) = b_i$ El numero de 1 en el contador luego de la operación i

t_i = cantidad de bits reseteados a 0

6	5	4	3	2	1	0

Vemos que:

El costo real de la operación es $Ci \leq t_i + 1$ (cuando paso todos los bits a 0 es t_i)

Si $b_i = 0 \rightarrow$ la operación i resetea los k bits a 0. Entonces $b_{i-1} = t_i = k$

Si $b_i > 0 \rightarrow b_i = b_{i-1} - t_i + 1$

Tenemos que: $b_i \leq b_{i-1} - t_i + 1$

i	b	t	C	
1	1	0	1	1
2	1	1	2	10
3	2	0	1	11
4	1	2	3	100
5	2	0	1	101

Contador binario - Potential method

Definimos:

$\Phi(D_i) = b_i$ El numero de 1 en el contador luego de la operación i

t_i = cantidad de bits reseteados a 0

Vemos que:

El costo real de la operación es $Ci \le t_i + 1$ (cuando paso todos los bits a 0 es t_i)

Si $b_i = 0 \rightarrow$ la operación i resetea los k bits a 0. Entonces $b_{i-1} = t_i = k$

Si $b_i > 0 \rightarrow b_i = b_{i-1} - t_i + 1$

Tenemos que: $b_i \le b_{i-1} - t_i + 1$

$B_{i-1} = 3$

0	**0**	**1**	**0**	**0**	**1**	**1**
6	5	4	3	2	1	0

$T_i = 2$

$B_i = 2$

0	**0**	**1**	**0**	**1**	**0**	**0**
6	5	4	3	2	1	0

$B_i \le B_{i-1} - t_i + 1$

$2 \le 3 - 2 + 1$

Contador binario - Potential method (cont.)

Finalmente vemos que:

$$\Phi(D_i)-\Phi(D_{i-1})\leqslant(b_{i-1}-t_i+1)-b_{i-1}=1-t_i$$

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

Y tenemos que

$\Phi(D_0)=0$ y $\Phi(D_i)\geq 0$ para todo $i>0$

Por lo tanto $\sum_{i=1}^{n}\hat{C}_i$ es O(n) y el costo de una operación amortizadas es O(1)