Análisis amortizado: Teoría y Ejemplo

Teoría de Algoritmos I (75.29 / 95.06)

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

Motivación

- Cuando analizamos un algoritmo utilizamos como medida su complejidad asintótica.
- Evaluamos para cada operación su “peor caso”
- Existen situaciones donde esto nos retornará un análisis incorrecto.

¿Qué es?

- Técnica para analizar el tiempo de ejecución de un algoritmo.
- Determina el rendimiento promedio de cada operación en el peor de los casos
- Apropiada cuando lo que interesa entender es el comportamiento asintótico de una secuencia de operaciones
- Puede mostrar que – aún existiendo algunas operaciones costosas – en promedio el costo es pequeño

¿Cuándo se originó?

- **Presentado en 1985 como una técnica de análisis de algoritmos.**

  En el paper "AMORTIZED COMPUTATIONAL COMPLEXITY" por Robert Endre Tarjan".

- **Engloba varios métodos**

  Acredita a diferentes autores por su invención

Métodos

- **Aggregate analysis**

  Aho, A. V., J. E. Hopcroft, and J. D. Ullman. 1974. The design and analysis of computer algorithms

- **Accounting method**

  Brown, M. R., and R. E. Tarjan. 1980. “Design and analysis of a data structure for representing sorted lists.” SIAM Journal on Computing

- **Potential method**

  D. Sleator (1983?)

Aggregate analysis

Demuestra que, para todo n, una secuencia de n operaciones requiere un tiempo total en el peor de los casos de T(n)

Se debe acotar a aquellas secuencias posibles de operaciones por la naturaleza racional de su uso.

Por lo tanto, el costo amortizado por operación es T(n) / n

A todas las operaciones – no importa si son diferentes – le asigna el mismo costo amortizado.

Aggregate analysis: PILA

Tenemos la estructura "PILA", que contiene las siguientes operaciones:

PUSH (x): Agrega a la pila el elemento x → O(1)

POP(): Extrae un elemento de la pila → O(1)

MULTIPOP(k): Extrae k elementos → O(k), O(n) si k>=n

```
Mientras haya elementos en pila y K>0
    pop()
    k--
```

Aggregate analysis: PILA (cont.)

Cual es el peor costo posible de n operaciones?

El peor costo de una operación es de multipop $O(n)$

Puedo realizar n multipop? → $n^* O(n) = O(n^2)$ **NO!!**

Para realizar n pop, primero se deben realizar n push

#pop + #multipops ≤ #push

Por lo tanto en el peor de los casos puedo hacer "n-1" push y 1 multipop de "n-1" elementos.

$(n-1)^*O(1) + 1^*O(n-1) \rightarrow O(n)$

Por lo tanto T(n) = n

Y el costo amortizado de cada operación $T(n)/n \rightarrow O(1)$

Accounting method

Este método es conocido también como "el método del banquero"

Se asignan diferentes costos a las diferentes operaciones

- Algunas con valor mayor y otras menor al costo real (C_i)

El costo de la operación asignado se conoce como "costo amortizado" ($\hat{C}_i$)

- Si el costo amortizado es mayor al real → la diferencia es un crédito
- Este crédito se puede utilizar para pagar futuras operaciones con costo real mayor a su costo amortizado.

Accounting method (cont.)

Usaremos al costo amortizado como la cota superior del costo real

Pediremos que para toda operación “n” cumpla con la cota:

$$\sum_{i=1}^{n} \hat{C}_i \geqslant \sum_{i=1}^{n} C_i$$

Por lo tanto el crédito para cualquier “n” no debe ser negativo

Y tenemos el costo amortizado como cota:

$$T(n) = \sum_{i=1}^{n} \hat{C}_i$$

Accounting method: PILA

Los costos reales de las operaciones son:

PUSH → 1

POP → 1

MULTIPOP → min (k, #S)

Proponemos los siguientes costos amortizados:

PUSH → 2

POP → 0

MULTIPOP → 0

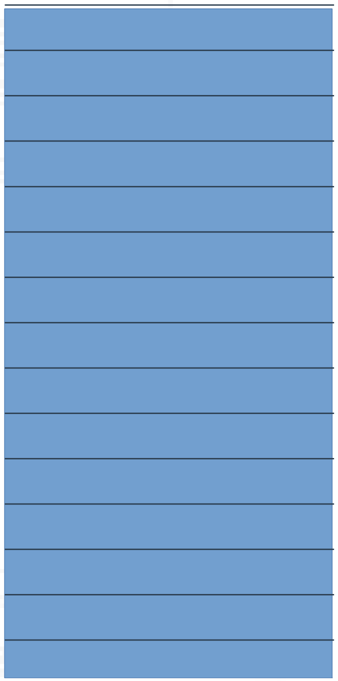
Accounting method: PILA (cont.)

El costo del PUSH "paga" su costo y el de un futuro POP (o multipop)

- PUSH → 1
- POP → 1
- MULTIPOP → min (k, #S)

- PUSH → 2
- POP → 0
- MULTIPOP → 0

	$\Sigma\hat{C}_i$	ΣC_i
PUSH	2	1
PUSH	4	2
POP	4	3
PUSH	6	4
PUSH	8	5
MULTIPOP (3)	8	8
PUSH	10	9
PUSH	12	10
POP	12	11



PILA

Accounting method: PILA (cont.)

Finalmente:

$$T(n) = \sum_{i=1}^{n} \hat{C}_i$$

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

Potential method

El trabajo prepagado se representa como “energia potencial”

Con eso se pagan operaciones futuras

El potencial esta asociado a toda la estructura de datos (y no a objectos especificos dentro de ella)

Llamaremos:

C_i el costo real de la operación i-esima

D_i la estructura de datos resultante de aplicar la operación i-esima a D_{i-1}

$\Phi(D_i)$ es la funcion de potencial que asigna un numero real a D_i

$$\mathbf{D_0} \xrightarrow[C_1]{Op_1} \mathbf{D_1} \xrightarrow[C_2]{Op_2} \mathbf{D_2} \xrightarrow[C_3]{Op_3} \cdots\cdots \xrightarrow[C_n]{Op_n} \mathbf{D_n}$$

$$\Phi(D_0) \qquad \Phi(D_1) \qquad \Phi(D_2) \qquad \qquad \Phi(D_n)$$

Potential method (cont.)

Definimos:

El costo amortizado de la operación i-esima como:

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

Podemos calcular el costo amortizado total de n operaciones:

$$\sum_{i=1}^{n} \hat{C}_i = \sum_{i=1}^{n} \left(C_i + \Phi(D_i) - \Phi(D_{i-1})\right)$$

$$= \sum_{i=1}^{n} (C_i) + \Phi(D_n) - \Phi(D_0)$$

Potential method (cont.)

Si podemos definir Φ tal que $\Phi(D_n) \geq \Phi(D_o)$

Entonces el costo amortizado $\sum_{i=1}^{n} \hat{C}_i$ nos sirve como cota superior del costo real $\sum_{i=1}^{n} C_i$

Pedimos que $\Phi(D_i) \geq \Phi(D_o)$ para todo i.

Entonces garantizamos el "pago en adelanto" (como en el metodo del banquero)

Usualmente se define $\Phi(D_0)=0$ y se prueba que $\Phi(D_i)\geq 0$ para toda operación i

Potential method: PILA

Definimos la función potencial como la cantidad de elementos en la pila.

$\Phi(D_0)=0$ (inicialmente hay 0 elementos en la pila)

$\Phi(D_i)\geq 0$ (no pueden existir cantidad negativo de elementos)

Calculamos el costo costo de las diferentes operaciones de la pila:

PUSH:

$$C_i=1$$

$$\Phi(D_i)-\Phi(D_{i-1})=(s+1)-s=1$$

$$\hat{C}_i=C_i+\Phi(D_i)-\Phi(D_{i-1})=1+1=2$$

Potential method: PILA (cont.)

POP:

$$C_i = 1$$

$$\Phi(D_i) - \Phi(D_{i-1}) = s - (s+1) = -1$$

$$\hat{C}_i = C_i + \Phi(D_i) - \Phi(D_{i-1}) = 1 - 1 = 0$$

MULTIPOP:

$$C_i = k$$

$$\Phi(D_i) - \Phi(D_{i-1}) = s - (s+k) = -k$$

$$\hat{C}_i = C_i + \Phi(D_i) - \Phi(D_{i-1}) = k - k = 0$$