Diseño y Análisis de Algoritmos
Análisis de Algoritmos

Dr. Jesús Ariel Carrasco Ochoa
ariel@inaoep.mx
Oficina 8311

Contenido

- Análisis de Algoritmos
- Análisis de Loops
- Análisis de Algoritmos recursivos
- Barómetros
- Análisis Amortizado

Análisis de Algoritmos

Estimar los recursos que un algoritmo requiere para ejecutarse

- Tiempo
  - número de operaciones (sumas, intercambios, etc.)
- Espacio
  - número de unidades de almacenamiento (bytes, palabras, etc.)

Análisis de Algoritmos

La estimación se hace relativa al tamaño del problema

La forma de medir el tamaño de un problema depende del problema específico

- Ordenar
  - Número elementos a ordenar n
- Multiplicación de matrices
  - Dimensiones de las matrices a multiplicar nxk y kxm

Análisis de Algoritmos

En algunos casos no es obvio

- Calcular el n-esimo número primo
  - ?
- Calcular la forma normal disyuntiva reducida de una función Booleana
  - ?
- Obtener las raíces de un polinomio de grado 2
  - ?

Análisis de Algoritmos

En algunos casos no es posible determinar exactamente el tiempo y espacio que requiere un algoritmo.

Algunas veces esto depende no solamente del tamaño del problema sino de la forma del mismo

- Calcular todos los reductos de un sistema de decisión
  - En algunos casos para sistemas de decisión del mismo tamaño encontrar todos los reductos puede tomar tiempo y espacio muy diferentes usando un mismo algoritmo

Análisis de Algoritmos

En estos casos se analizan por separado:

- Peor caso
  - Es una cota superior, nunca tardará más o requerirá más espacio
- Mejor caso
  - Es una cota inferior, nunca tardará menos o requerirá menos espacio
- Caso promedio
  - Es el tiempo/espacio esperado

Análisis de Algoritmos

El caso promedio no es:

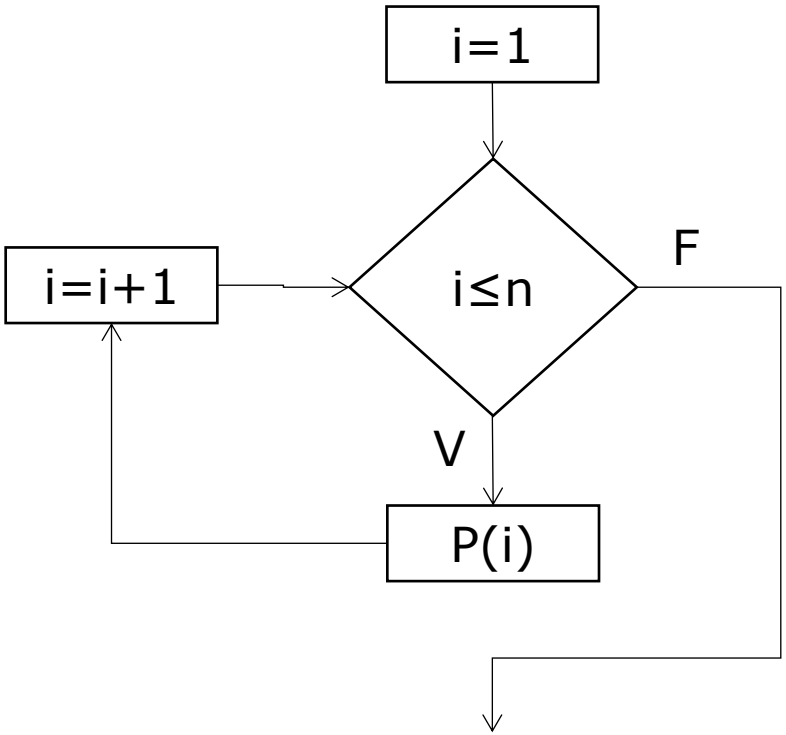
(Mejor-caso + Peor-caso) / 2

El caso promedio tiene que ver con el tiempo/espacio de cada caso posible y la probabilidad de que se presente cada caso.

- Usualmente calcular el caso promedio es muy difícil
- El mejor caso no es útil
- Comúnmente se calcula el peor caso

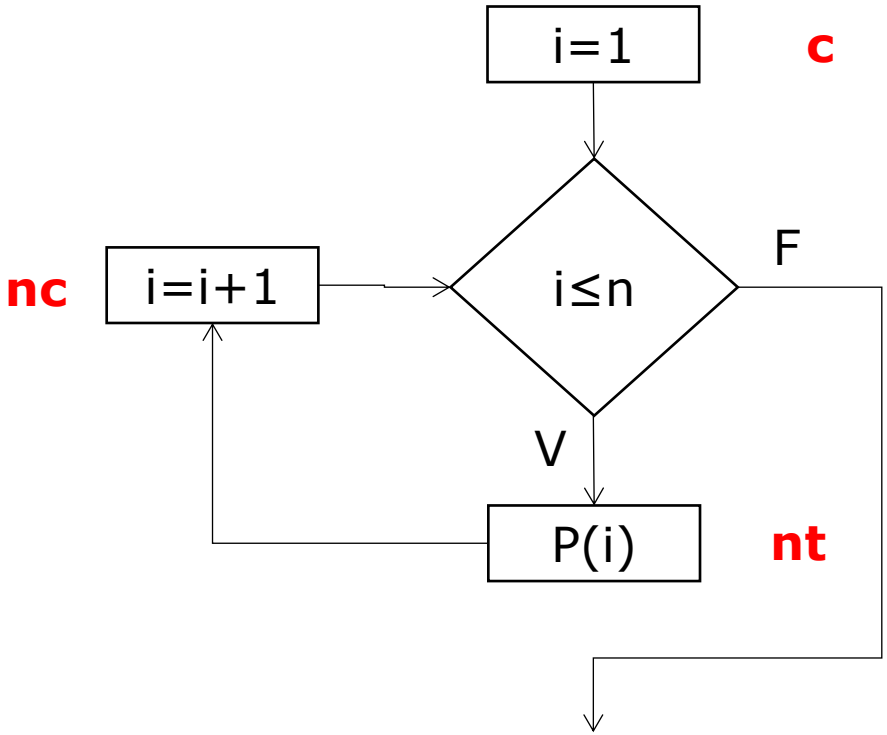
Análisis de Loops

Dado un Loop:



Análisis de Loops

Dado un Loop:

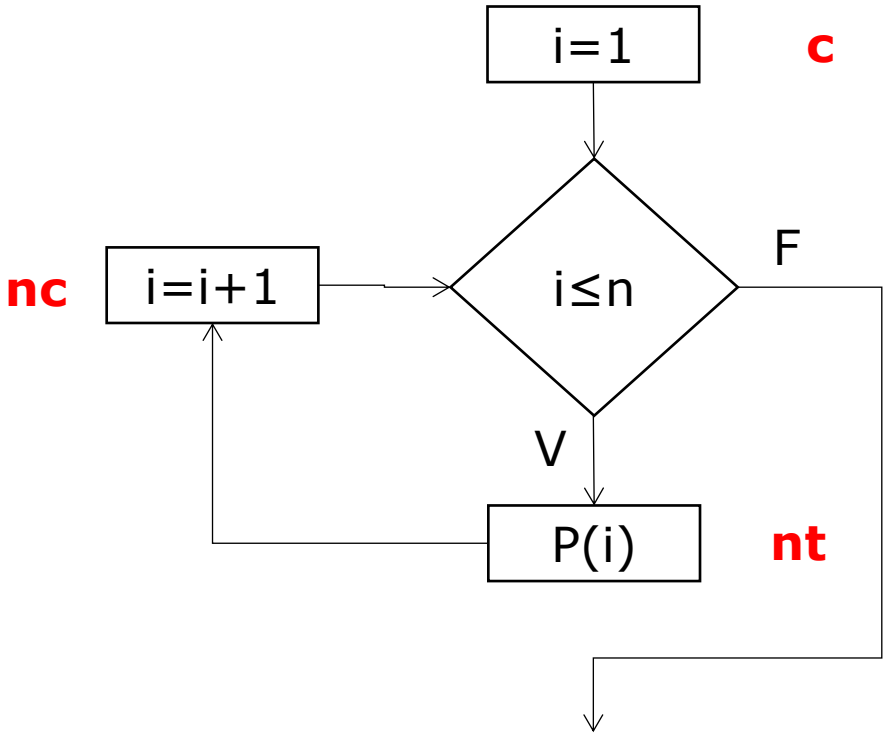


Siendo t el tiempo de P(i)

Análisis de Loops

Dado un Loop:

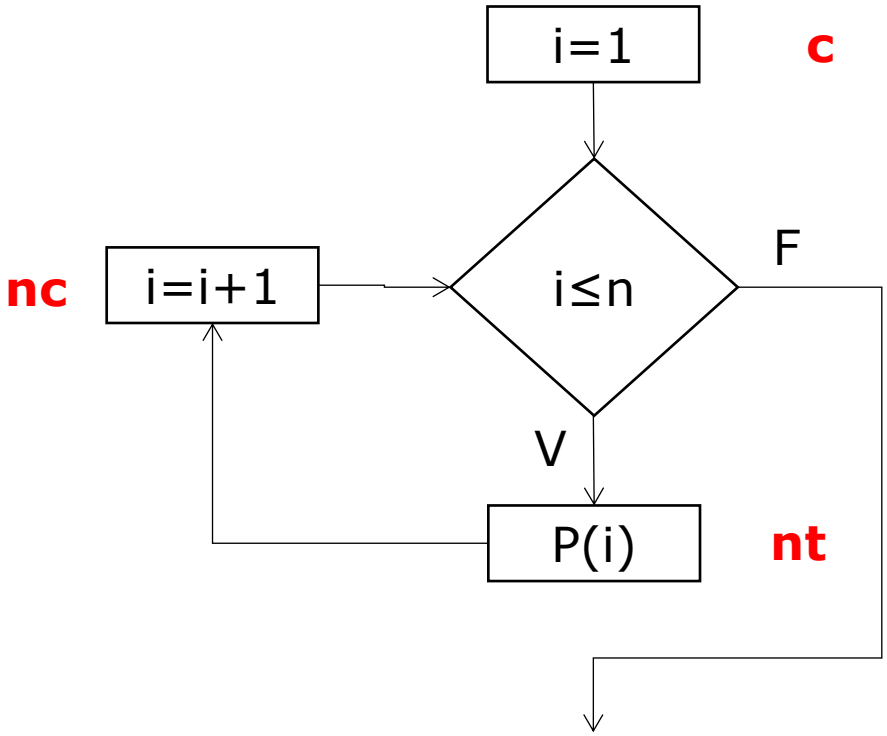
En total (t+3c)n+2c



Siendo t el tiempo de P(i)

Análisis de Loops

Dado un Loop:



En total (t+3c)n+2c

O(nt) si t depende de o de n

O(n) si t es constante

(n+1)c

Siendo t el tiempo de P(i)

Ejemplo: Números de Fibonacci

```
Entrada: n el número a generar

Salida:  fn el n-ésimo número de Fibbonaci

ti=0
tj=1
for( i=1 ; i<=n ; i++)
   {
       tj = tj+ti;
       ti = tj-ti;
   }
fn = tj
```

O(1)

O(n)

Ejemplo: Ordenamiento de Burbuja

```
Entrada: a un arreglo de números
         n el tamaño del arreglo

Salida:  a ordenado
         Para todo i=1,…,n-1 a[i]<=a[i+1]
```

```
for( i=1 ; i<n ; i++)
      for( j=0 ; j<n ; j++)
            if( a[j] > a[j+1] )
              {
                temporal = a[j+1];
                a[j+1] = a[j];
                a[j] = temporal;
              }
```

O(1) **O(n)** **O(n^2)**

Análisis de Algoritmos Recursivos

Dado un Algoritmo recursivo la complejidad se plantea como una recurrencia siguiendo la forma del algoritmo

Nota: la recurrencia debe representar al algoritmo no a lo que calcula

Ejemplo: Números de Fibonacci

```
Entrada: n el número a generar

Salida:  fn el n-ésimo número de Fibbonaci

If( n ≤ 2 )
    fn = 1;
else
    fn = fib(n-1)+fib(n-2)
```

$T(1) = 1$

$T(2) = 1$

$T(n) = T(n-1)+T(n-2)+2$

Ejemplo: Factorial

```
Entrada: n el factorial a generar

Salida:  fn el factorial de n

If( n == 0 )
    fn = 1;
else
    fn = n*fact(n-1)
```

$T(0) = 1$

$T(n) = T(n-1)+2$

Uso de Barómetros

En ocasiones es muy complicado contar todas las operaciones pues dado que pueden tener costos diferentes

Se deben contar por separado y el tiempo final depende del costo de cada tipo de operación

Uso de Barómetros

Dado que lo que se quiere acotar asintóticamente el tiempo no es necesario contar todas las operacioes

Se selecciona una que sirva como barómetro para determinar el costo del algoritmo.

Uso de Barómetros

Como barómetro se selecciona la operación que más se repite.

Si hay varias operaciones que se repiten aproximadamente el mismo número de veces se selecciona la más costosa

Ejemplo: Ordenamiento de Burbuja

```
Entrada: a un arreglo de números
         n el tamaño del arreglo

Salida:  a ordenado
         Para todo i=1,…,n-1 a[i]<=a[i+1]

for( i=1 ; i<n ; i++)
      for( j=0 ; j<n ; j++)
            if( a[j] > a[j+1] )
              {
                temporal = a[j+1];
                a[j+1] = a[j];
                a[j] = temporal;
              }
```

Barómetro

Análisis Amortizado

En ocasiones en un Loop las repeticiones no tienen todas el mismo costo

Por esto el análisis debe hacerse con más cuidado

Ejemplo: Ordenamiento de Burbuja

```
Entrada: a un arreglo de números
         n el tamaño del arreglo

Salida:  a ordenado
         Para todo i=1,…,n-1 a[i]<=a[i+1]
```

```
for( i=1 ; i<n ; i++)
      for( j=0 ; j<(n-i) ; j++)
            if( a[j] > a[j+1] )
              {
                temporal = a[j+1];
                a[j+1] = a[j];
                a[j] = temporal;
              }
```

O(1)

Análisis Amortizado

En este caso el Loop interno se repite n-i veces para cada valor de i desde 1 hasta n-1, entonces el total de repeticiones es:

$$\sum_{i=1}^{n-1} n - i = \sum_{i=1}^{n-1} i = \frac{n(n-1)}{2} = O(n^2)$$

Ejemplo: Ordenamiento de Burbuja

```
Entrada: a un arreglo de números
         n el tamaño del arreglo

Salida:  a ordenado
         Para todo i=1,…,n-1 a[i]<=a[i+1]
```

```
for( i=1 ; i<n ; i++)
      for( j=0 ; j<(n-i) ; j++)
            if( a[j] > a[j+1] )
               {
                 temporal = a[j+1];
                 a[j+1] = a[j];
                 a[j] = temporal;
               }
```

$O(1)$

$n - i = O(n)$

$O\left(\frac{n(n-1)}{2}\right) = O(n^2)$

Análisis Amortizado Ejemplo 2

```
Entrada: n el número de repeticiones por Loop

Salida:  Resultado

for( j=0 ; j<n ; j++)
      for( i=0 ; i<n ; i++)
            P(i)
```

Donde:

P(i) tiene costo i si i es una potencia de 2 y 1 en otro caso

Análisis Amortizado Ejemplo 2

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

Análisis Amortizado Ejemplo 2

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

A lo más $log_2(n)$

Análisis Amortizado Ejemplo 2

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

A lo más $log_2(n)$, entonces

$$\sum_{i=1}^{log_2(n)} 2^i + \sum_{i=1}^{n-log_2(n)} 1 = \frac{2^{log_2(n)+1} - 1}{2-1} + n - log_2(n)$$

$$= 2n - 1 + n - log_2(n)$$
$$= 3n - log_2(n) - 1$$
$$= O(n)$$

Análisis Amortizado Ejemplo 2

```
Entrada: n el número de repeticiones por Loop

Salida:  Resultado

for( j=0 ; j<n ; j++)
      for( i=0 ; i<n ; i++)
            P(i)
```

O(1) **O(n)** **$O(n^2)$**

Donde:

P(i) tiene costo i si i es una potencia de 2 y 1 en otro caso

Análisis Amortizado Ejemplo 3

```
Entrada: n el número de repeticiones por Loop

Salida:  Resultado

for( j=0 ; j<n ; j++)
      for( i=0 ; i<n ; i++)
            P(i)
```

Donde:

P(i) tiene costo n si i es una potencia de 2 y 1 en otro caso

Análisis Amortizado Ejemplo 3

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

Análisis Amortizado Ejemplo 3

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

A lo más $log_2(n)$

Análisis Amortizado Ejemplo 3

En este caso debemos considerar cuántas potencias de 2 se presentan en las n repeticiones ?

A lo más $log_2(n)$, entonces

$$\sum_{i=1}^{log_2(n)} n + \sum_{i=1}^{n-log_2(n)} 1 = nlog_2(n) + n - log_2(n)$$

$$= O\left(nlog_2(n)\right)$$

Análisis Amortizado Ejemplo 3

```
Entrada: n el número de repeticiones por Loop

Salida:  Resultado

for( j=0 ; j<n ; j++)
      for( i=0 ; i<n ; i++)
            P(i)
```

$O(\log_2(n))$ $O(n\log_2(n))$ $O(n^2\log_2(n))$

Donde:

P(i) tiene costo n si i es una potencia de 2 y 1 en otro caso

Diseño y Análisis de Algoritmos
Análisis de Algoritmos

Dr. Jesús Ariel Carrasco Ochoa
ariel@inaoep.mx
Oficina 8311