Análisis y Diseño de Algoritmos

Hernán Gómez

Universidad del Valle
(Facultad de Ingeniería)

Febrero, 2020

Diseño de Algoritmos

- El Diseño se refiere a la búsqueda de métodos o procedimientos, conjunto finito de instrucciones adecuadas al dispositivo que disponemos.
- En el diseño prima la idea de las sencillez en el código sin olvidar la eficiencia de recursos que debe tener.
- El diseño de algoritmos en independiente de la maquina donde se ejecuta.

Universidad del Valle

Eficiencia Algoritmica

La eficiencia se refiere a:

- **Eficiencia en memoria o complejidad espacial**
- **Eficiencia en tiempo o complejidad temporal**

Eficiencia Temporal

La eficiencia temporal depende de:

- Los datos de entrada
- Calidad del código objeto generado por el compilador.
- Rapidez del procesador donde se ejecuten las instrucciones
- Diseño adecuado del algoritmo(secuencia o paralelo)

Estudio temporal en Algoritmos

Estudio teórico(a priori): estudio teórico matemático que consiste en obtener una función que acote (por arriba o por abajo) el tiempo de ejecución del algoritmo para unos valores de entrada dados.

Principio de Invarianza

Dado un algoritmo y dos implementaciones I_1 e I_2, que tardan $T_1(n)$ y $T_2(n)$

- El Principio de Invarianza afirma que existe una constante real $c > 0$ y un número natural n_0 tales que para todo $n \geq n_0$ se verifica que $T_1(n) \leq cT_2(n)$.
- El tiempo de ejecución de dos implementaciones distintas de un algoritmo dado no va a diferir más que en una constante multiplicativa.

Parámetros para la Complejidad Temporal

- La constante multiplicativa (C).
- El n_0, valor inicial de n; el n hace referencia al tamaño de las entradas.

Ejemplo Comparación entre dos Algoritmos

Ejemplo: dos algoritmos cuyos tiempos de ejecución son: algoritmo A:$106n^2$ segundos y algoritmos B: $5n^3$ segundos el primero sólo será mejor que el segundo para tamaños de la entrada superiores a 21.

Ejemplo

Problema:Realizar un algoritmo que calcule la traza de una matriz.
Definición: La traza de una matriz es la suma de los elementos de la diagonal principal

1	2	1
0	5	7
8	3	4

Cuadro: Traza de matriz cuadrada M_{3x3}: 1+5+4=10

Algoritmo I

```
Real suma <-0;

Desde i<-1; i<=n; i<- i+1
  {
   Desde j<-1; j<=n; j<-j+1
    {
      si (i=j)
      {
       suma<-suma +datos[i][j];
       }
    }
  }

Imprimir (suma)
```

Algoritmo II

```
Real suma <-0;

Desde i<-1; i<=n; i<- i+1
  {
        suma<-suma +datos[i][i];

    }
  }

Imprimir (suma)
```

Cuestiones?

- El algoritmo#1 es correcto?
- El algoritmo#2 es correcto?
- Cual de los algoritmos gasta mas recursos en memoria?
- Cual se ejecuta mas rápido?

Porque Estudiar FADA?

Por que provee muchas herramientas para temas importantes como:

- Ordenamiento.
- Estructura de datos: pilas, colas, montículos, arboles entre otros.
- Búsquedas.

Provee una metodologías para el diseño de algoritmos

- Divide y conquistaras.
- Voraces.
- Programación dinámica.
- Algoritmos aleatorios.

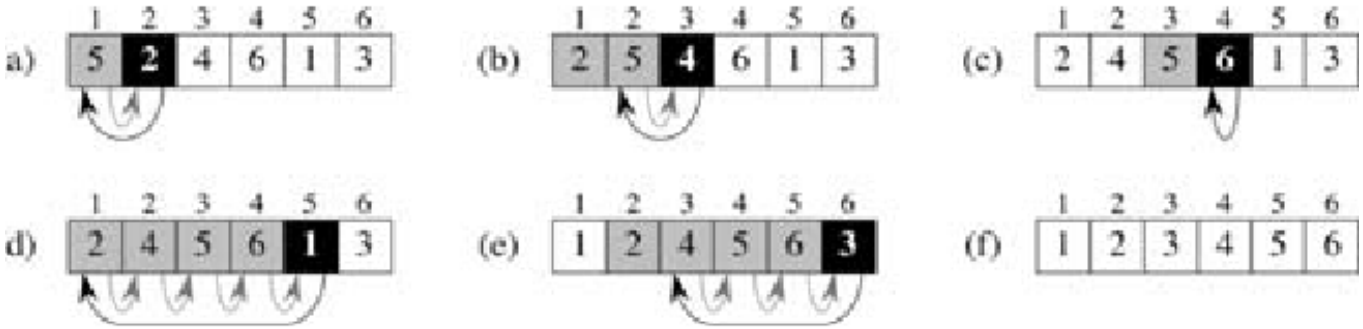
Abstracciones

- grafos:Búsqueda de caminos óptimos en una red y ordenamiento calendario clases

Como se Comparan Algoritmos

- Experimentación del programa variando parámetros, entradas y verificando correctitud.
- Obteniendo la ejecución promedio.
- Obteniendo el peor de los casos.

Ejemplo Insertion Sort



Algoritmo Insertion

INSERTION-SORT(A, n)	*cost*	*times*
for $j = 2$ **to** n	c_1	n
$key = A[j]$	c_2	$n-1$
// Insert $A[j]$ into the sorted sequence $A[1..j-1]$.	0	$n-1$
$i = j-1$	c_4	$n-1$
while $i > 0$ and $A[i] > key$	c_5	$\sum_{j=2}^{n} t_j$
$A[i+1] = A[i]$	c_6	$\sum_{j=2}^{n}(t_j - 1)$
$i = i-1$	c_7	$\sum_{j=2}^{n}(t_j - 1)$
$A[i+1] = key$	c_8	$n-1$

- El pseudocodigo no es un programa es solo un boceto del funcionamiento de la lógica.
- Muchos detalles solo pueden ser visto en la ejecución del programa.
- La correctitud en tiempo de ejecución es indispensable. No basta con el modelo teórico.

Análisis

- Verificación de exactitud y terminación.
- Depende del tamaño y las propiedades de la entrada.
- Siempre se desean cotas, limites inferiores o superiores.

Analisis

Las cotas inferiores y superiores proporcionan una idea del tiempo de ejecución con respecto a la entrada (n)

- Análisis considerados:

1. Mejor caso
2. Caso promedio
3. Peor caso. máximo tiempo de ejecución para cualquier entrada **max** $T(n)$.

El caso promedio, junto con el peor caso son los de mayor estudio (casi todas las entrada son abordadas)