Análisis de algoritmos

Simplicidad: facilita su verificación, el estudio de su eficiencia y su mantenimiento

Eficiencia: Cantidad de recursos, principalmente de memoria y tempo, que necesita para ejecutarse

Estudio de eficiencia

* Permite medir el costo, en tiempo, memoria u otro recurso que consume un algoritmo para encontrar la solución a un problema
* Ofrece la posibilidad de comparar distintos algoritmos que resuelven un mismo problema

**Factores** de los cuales depende el **tiempo de ejecución**

* Datos de entrada (factor principal a analizar)
* La calidad del código generado por el compilador para crear el programa objeto
* La naturaleza y la rapidez de las instrucciones de maquina del procesador concreto que ejecute el programa
* La complejidad intrínseca del algoritmo

Para analizar la complejidad de un algoritmo y caracterizar su costo existen DOS ESTUDIOS:

* El 1ro ofrece una medida empírica(a posteriori), consiste en medir el tiempo de ejecución del algoritmo para unos valores de entrada dados en un ordenador concreto
* El 2do proporciona una medida teórica(a priori) que consiste en determinar matemáticamente el tiempo de ejecución de un algoritmo como función del tamaño de datos de entrada

Reglas para el calculo de unidades de tiempo de instrucciones

1\_ las declaraciones no consumen tiempo

2\_ sentencias simples: 1ut

3\_expresiones aritméticas o relacionales: 1ut

4\_clases incondicionales

5\_

6\_

7\_

T(n) -> tiempo de ejecución de un algoritmo: numero de operaciones ejecutadas por un ordenador idealizado para una entrada de tamaño n

**Principio de invarianza**

Dado un algoritmo, y dos implementaciones I1 y I2, que tardan t1(n) y t2(n) segundos para resolver un caso de tamaño n, entonces siempre que existen constantes positivas c y d tales que:

T1(n) <= c\*t2(n) y t2(n)<= d\* T1(n)

El tiempo de ejecución de un algoritmo va a ser una función que mida el numero de operaciones elementales que realiza el algoritmo para un tamaño de entrada dado

Suele estudiarse tres casos:

* Caso mejor: mínimo valor de t(n) para entradas de tamaño n
* Caso peor: máximo valor de t(n) para entradas de tamaño n
* Caso medio: valor medio del tiempo de ejecución de todas las entradas de tamaño n

Medidas asintóticas

Es un conjunto de funciones que muestran un comportamiento similar cuando los argumentos toman valores muy grandes.

En análisis de algoritmos una **cota ajustada asintótica** es una función que sirve de cota de otra función, tanto superior como inferior, cuando el argumento tiende a infinito.

Las notaciones asintóticas son lenguajes que nos permiten analizar el tiempo de ejecución de un algoritmo identificando su comportamiento si el tamaño de entrada para el algoritmo aumenta. Esto también se conoce como la **tasa de crecimiento** de un algoritmo

Propósito: caracterizar el costo de un algortimo mediante funciones simples, que acoten superior e inferior mente el costo de toda instancia, para n suficientemente grandes