Instituto Politécnico Nacional
Escuela Superior de Cómputo

Analisis de algoritmos, Sem: 2021-1, 3CV1,Práctica 2, 27/10/2020

Práctica 2: Funciones Recursivas vs Iterativas

Bejar Valencia Angel Ivan,Payán Téllez René
ivanbejar07@gmail.com,rpayant1500@alumno.ipn.mx

Resumen: En esta practica se compararan la complejidad de los algorimos iterativos y recursivos
Palabras clave: Iterativo,Complejidad,Recursivo,C/C++

1 Introduccion

Durante esta práctica, se analizará y comparará la complejidad de soluciones iterativas y recursivas para el mismo algoritmo. La diferencia entre una solución iterativa y una solución recursiva es que, en una solución recursiva, se invoca la respuesta de múltiples invocaciones de la misma función, para obtener una respuesta a partir de la división del problema en partes mas pequeñas. Mientras que, en una solución iterativa, no se hace uso de dicho recurso, todo el algoritmo se ejecuta sin necesidad de invocar a una función desde sí misma. Por lo que se pueden realizar soluciones al mismo problema de ambas formas y estas a su vez podrian tener distintas complejidades.

2 Conceptos Basicos

2.1 Algoritmo

La palabra algoritmo proviene del sobrenombre de un matemático árabe del siglo IX, Al-Khwarizmi, que fue reconocido por enunciar paso a paso las reglas para las operaciones matemáticas básicas con decimales (suma, resta, multiplicación y división). Vemos definición de algoritmo como un grupo de órdenes consecutivas que presentan una solución a un problema o tarea. Algunos ejemplos de algoritmos los podemos encontrar en las matemáticas (como el algoritmo para resolver una multiplicación) y en los manuales de usuario de un aparato (como una lavadora o una impresora). Sin embargo, hoy en día se relaciona la palabra algoritmo con el mundo de la informática, más concretamente en la programación; los conocidos como algoritmos informáticos.[1]

2.2 Complejidad algoritmica

Así que, por su naturaleza, un problema tiene la capacidad de ser solucionado por uno o varios métodos, pero si bien es importante llegar a la respuesta, más importante es evaluar su viabilidad. Siempre que se analiza y evalúa adecuadamente la efectividad de una solución, disminuye drásticamente el costo que representa su producción y mantenimiento, pues los recursos que se invierten posteriormente en codificación, pruebas y revisión es mucho menor siempre (como el tiempo, dinero y talento humano). Entrando en materia, la complejidad algorítmica es una métrica teórica que nos ayuda a describir el comportamiento de un algoritmo en términos de tiempo de ejecución (tiempo que tarda un algoritmo en resolver un problema) y memoria requerida (cantidad de memoria necesaria para procesar las instrucciones que solucionan dicho problema). Esto nos ayuda a comparar entre la efectividad de un algoritmo y otro, y decidir cuál es el que nos conviene implementar.[2]

2.3 Algoritmo Iterativo

Las instrucciones de repetición, de iteración o bucles, facilitan la repetición de un bloque de instrucciones, un número determinado de veces o mientras se cumpla una condición. Por lo general, existen dos tipos de estructuras iterativas o bucles en los lenguajes de programación. Encontraremos un tipo de bucle que se ejecuta un número preestablecido de veces, que es controlado por un contador o índice, incrementado en cada iteración. Este tipo de bucle forma parte de la familia for. Por otro lado, encontraremos un tipo de bucle que se ejecuta mientras se cumple una condición. Esta condición se comprueba al principio o el final de la construcción. Esta variante pertenece a la familia while or repeat, respectivamente.[2]

2.4 Algoritmo Recursivo

Los algoritmos recursivos se basan en la metodología de llamar repetidamente la propia función en que están definidos, y son de gran utilidad en multitud de campos en la informática.[3]
La solución de un algoritmo recursivo comienza a atacar un problema tomando un problema grande y lo va dividiendo hasta llegar a problemas más pequeños que son sencillos o triviales de resolver. Una vez que llega a estos problemas más pequeños, vuelve por donde llegó y comienza a combinar las soluciones entre si para obtener la solución al problema más grande.[4]

3 Experimentacion y Resultados

4 Conclusiones

5 Anexo

6 Bibliografia

(1)https://openwebinars.net/blog/que-es-un-algoritmo-informatico/ (2)https://medium.com/@joseguillermo$_{/}qu$(3)$https$
$//rsanchezs.gitbooks.io/ciencia-de-datos-con-r/content/estructuras_control/iterativas/estructuras_iterativas.htm$
$//formacion.desarrollando.net/cursosfiles/formacion/curso_454/deda-03.pdf(5)https://thatcsharpguy.com/tv/re$
$iteracion/$