INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

-------------------ANÁLISIS DE ALGORITMOS------------------

ACTIVIDAD

Análisis de algoritmos recursivos

PROFESOR:

Franco Martínez Edgardo Adrián

ALUMNO:

Meza Vargas Brandon David – 2020630288

GRUPO:
3CM13

Índice

Para los siguientes algoritmos considerarán las siguientes operaciones:

- Asignaciones y returns
- Aritméticas
- Comparación

Código 01

```
Busqueda(A[],i,val){
    if(i<0)
        return -1;
        if(A[i] == val)
            return i
        return Busqueda(A[],i-1,val);
}
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
Busqueda(A[],i,val){
    if(i<0)                                 ----- 1
        return -1;                          ----- 1
        if(A[i] == val)                     ----- 1
            return i                        ----- 1
        return Busqueda(A[],i-1,val);       ----- 2
}
```

Podemos ver que tenemos un primer caso en donde i es menor que 0, en este caso tenemos:

T(0)=2, la comparación y el return

En caso contrario tenemos

$$T(n) = 1+2´T(n-1)$$

De lo anterior tenemos la comparación del primer if, la comparación y return del segundo if cuando el valor es encontrado y se le suma la llamada a la misma función con una transformación, la cual es que se llama al mismo problema reducido en uno, de esta forma tenemos:

$$\mathbf{T(n)=3+T(n-1)}$$

Con el modelo recurrente anterior podemos hallar su equivalente evaluando para alguna n, en este ejemplo lo haremos para n=5 sabiendo que T(0)=2:

$$T(0)=2$$

$$T(1)=3+T(0)=3+2=5$$

$$T(2)=3+T(1)=3+5=8$$

$$T(3)=3+T(2)=3+8=11$$

$$T(4)=3+T(3)=3+11=14$$

$$T(5)=3+T(4)=17$$

Analizando el comportamiento anterior nos damos cuenta de que podemos hallar un modelo sin recurrencia y su cota:

$$T(n)=3n+2 \in O(n)$$

Código 02

```
int coef(int n, int k){
    if(k == 0 || k == n)
        return 1;
    else if(k > 0 && k< n)
        return coef(n-1,k-1)+coef(n-1,k);
}
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
int coef(int n, int k){
    if(k == 0 || k == n)                            ----- 1
        return 1;                                   ----- 1
    else if(k > 0 && k< n)                          ----- 2
        return coef(n-1,k-1)+coef(n-1,k);           ----- 5
}
```

Podemos ver que el caso base es donde k es igual a 0 o igual a n, en este caso tenemos lo siguiente:

T(0)=2 comparación y return

En caso contrario tenemos lo siguiente:

T(n)=1+2+5+T(n-1)+T(n-1) = 8+2T(n-1)

Lo anterior se obtuvo de la primera comparación, posteriormente las dos comparaciones que se hacen en el segundo if y las 5 operaciones que tenemos al final, además sumamos dos veces la llamada a la misma función pero reducida en 1.

Para calcular la cota de este algoritmo obtendremos un modelo no recurrente equivalente:

$$T(n) = 8 + 2(T(n-1))$$

$$T(n) - 2T(n-1) = 8$$

Sustituyendo

T(n) = x b = 1 d = 0

Así tenemos:

$$(x-2)(x-1) = 0$$

Obteniendo las raíces

r1 = 1 r2 = 2

Si T(0) = 2 y T(1) = 8+4 = 12

$$T(n) = c1(1)^n + c2(2)^n$$

$$T(0) = 2 = c1 + c2$$

$$T(1) = 12 = c1 + c2(2)$$

De esta forma calculando los coeficientes:

c1 = -8 c2 = 10

Por lo tanto:

$$T(n) = -8(1)^n + 10(2)^n \in O(2^n)$$

Código 03

```
Palindormo(cadena){
    if(longitud(cadena) == 1)
        return TRUE;
    if(primer_caracter(Cadena) != ultimo_caracter(cadena))
        return FALSE;

    cadena = remover_primer_ultimo_caracter(Cadena);
    Palindromo(cadena);
}
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
Palindormo(cadena){
    if(longitud(cadena) == 1)                                   ----- 1
        return TRUE;                                            ----- 1
    if(primer_caracter(Cadena) != ultimo_caracter(cadena))      ----- 1
        return FALSE;                                           ----- 1

    cadena = remover_primer_ultimo_caracter(Cadena);            ----- 1
    Palindromo(cadena);
}
```

Tenemos el caso base donde la longitud de la cadena es igual a 1, en esta parte tenemos:

$$T(0)=2 \text{ la comparación y el return}$$

Para los demás casos tenemos lo siguiente:

$$T(n)=1+1+1+T(n-1)=3+T(n-1)$$

Aquí contamos la primera comparación, la segunda comparación, la asignación a la cadena y la llamada transformada a la misma función con n-1, este -1 es por que se llama a la misma función haciendo una transformación a la cadena.

Aquí para calcular la cota pasamos a una función no recurrente de la siguiente forma:

$$T(0)=2$$

$$T(1)=3+T(0)=3+2=5$$

$$T(2)=3+T(1)=3+5=8$$

$$T(3)=3+T(2)=3+8=11$$

$$T(4)=3+T(3)=3+11=14$$

$$T(5)=3+T(4)=17$$

Analizando el comportamiento anterior nos damos cuenta de que podemos hallar un modelo sin recurrencia y su cota:

$$T(n)=3n+2 \in O(n)$$

Código 04

```
subAlgoritmo volados(n,cadena)
    Si n != 0
        volados(n-1,concatenar(cadena,'S'))
        volados(n-1,concatenar(cadena,'A'))
    SiNo
        Mostrar cadena
    FinSi
FinSubAlgoritmo
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
subAlgoritmo volados(n,cadena)
    Si n != 0                                            ----- 1
        volados(n-1,concatenar(cadena,'S'))              ----- 1
        volados(n-1,concatenar(cadena,'A'))              ----- 1
    SiNo
        Mostrar cadena
    FinSi
FinSubAlgoritmo
```

Tenemos un caso base el cual es:

T(0)=1, solo es la comparación del inicio.

Para lo siguiente tenemos que es:

T(n)=1+2+2T(n-1)=3+2(T(n-1))

En lo anterior contamos la comparación y las dos operaciones que están en la recursión, de igual forma contamos la transformación.

Para calcular la cota de este algoritmo obtendremos un modelo no recurrente equivalente:

$$T(n) = 3 + 2(T(n-1))$$

$$T(n) - 2T(n-1) = 3$$

Sustituyendo

T(n) = x b = 1 d = 0

Así tenemos:

$$(x-2)(x-1) = 0$$

Obteniendo las raíces

r1 = 1 r2 = 2

Si T(0) = 1 y T(1) = 3+2 = 5

$$T(n) = c1(1)^n + c2(2)^n$$

$$T(0) = 1 = c1 + c2$$

$$T(1) = 5 = c1(1) + c2(2)$$

De esta forma calculando los coeficientes:

c1 = -3 c2 = 4

Por lo tanto:

$$\boldsymbol{T(n) = -3(1)^n + 4(2)^n \in O(2^n)}$$

Código 05

```
decABin(n){
    if(n>1)
        DecABin(n/2)
        Mostrar(n%2)
}
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
decABin(n){
    if(n>1)             ----- 1
        DecABin(n/2)    ----- 1
        Mostrar(n%2)    ----- 1
}
```

Para el caso base donde n no sea mayor a 1, tendremos:

T(1)=1 donde solo se cuenta la comparación

Para los demás casos tendremos:

T(n)=1+2+T(n/2)=3+T(n/2)

De lo anterior se cuenta la comparación y las dos operaciones que se tienen dentro del if, además de la transformación que se hace.

En este algoritmo hacemos uso del **teorema maestro.**

De esta manera tenemos que

a=1 b=2 f(n)=3

Por lo tanto aplicando el límite asintótico tenemos:

$$O(n^{\log_2 1}) = O(n^0)$$

De esta manera aplicamos el segundo caso del teorema maestro ya que:

$$f(n) = O(n^0)$$

Así tenemos finalmente:

$$T(n) = O(n^0 \log(n))$$

$$T(n) = O(\log(n))$$

Código 06

```
int producto(int a, int b){
    if(b == 0)
        return 0;
    else
        return a + producto(a,b-1);

}
```

En la siguiente imagen podemos ver la cantidad de operaciones que tenemos por cada línea:

```
int producto(int a, int b){
    if(b == 0)                              ----- 1
        return 0;                           ----- 1
    else
        return a + producto(a,b-1);         ----- 3

}
```

Aquí tenemos el caso cuando b es igual a 0, en este caso tenemos:

$T(0)=2$, la comparación y el return

En caso contrario:

$$T(n)=1+3+T(n-1)=4+T(n-1)$$

De lo anterior se cuenta la primera comparación y las 3 operaciones dentro del else.

Así podemos encontrar una función no recurrente equivalente y de igual forma la cota haciendo algunas evaluaciones:

$$T(5)=4+T(4)=4+18=22$$

$$T(4)=4+T(3)=4+14=18$$

$$T(3)=4+T(2)=4+10=14$$

$$T(2)=4+T(1)=4+6=10$$

$$T(1)=4+T(0)=4+2=6$$

Analizando lo anterior llegamos a:

$$\boldsymbol{T(n)=4n+2 \in O(n)}$$