Cálculo del Tiempo de Ejecución - Algoritmos Recursivos

Temario

- Repaso del concepto de recursion.
- Introducción del concepto de función de recurrencia.
- Cálculo del T(n) para algoritmos recursivos.

Repaso: Recursion

Conceptos

- La recursión es una técnica de programación muy poderosa, en la cual una función realiza llamadas a misma en pos de resolver un problema.
- Razones para su uso:
  - Problemas “casi” irresolubles con las estructuras iterativas.
  - Soluciones elegantes.
  - Soluciones más simples.

Identificación de casos

- En las funciones recursivas bien definidas se puede identificar dos elementos:
  - **Caso Base**: Se da cuando el calculo es tan simple que se puede resolver directamente sin necesidad de hacer una llamada recursiva.
  - **Caso Recursivo**: aquí la función realiza algunas operaciones con las que se reduce la complejidad del problema y luego realiza un llamado a si misma.

Ejemplo de Recursión

```
#Calcula el Factorial.
factorial (n)
        if (n < 2)
                return 1
        else
                return n * factorial( n - 1 )
```

Caso Base: Cuando el parámetro es 1 o menor, la función retorna 1 y termina.

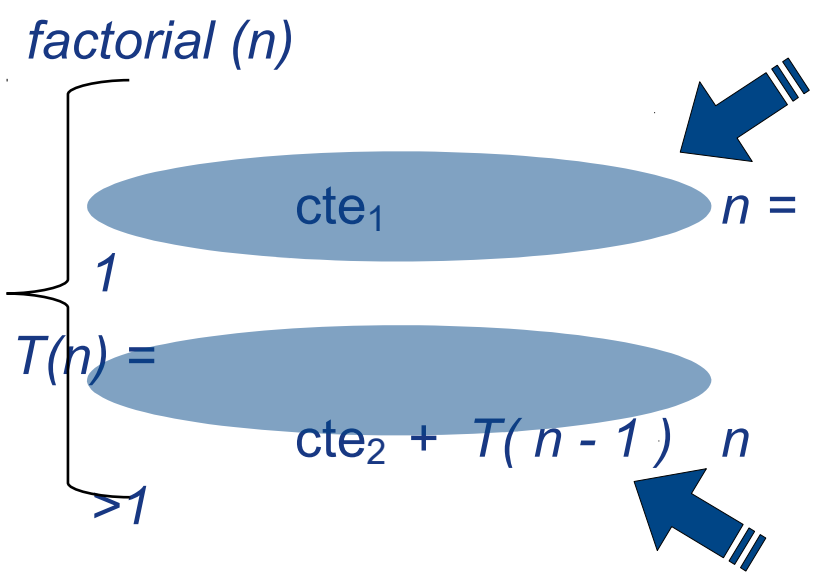
Caso Recursivo: Cuando el parámetro es distinto de 1, la función multiplica el parámetro con el resultado de volver a invocar a la función con el parámetro disminuido en 1.

Función de Recurrencia.

Cálculo del Tiempo de Ejecución - Algoritmos Recursivos

```
#Calcula el Factorial.
 factorial (n)
        if (n < 2)
                return 1
        else
                return n * factorial( n
    - 1 )
```

Cálculo del Tiempo de Ejecución - Función de recurrencia



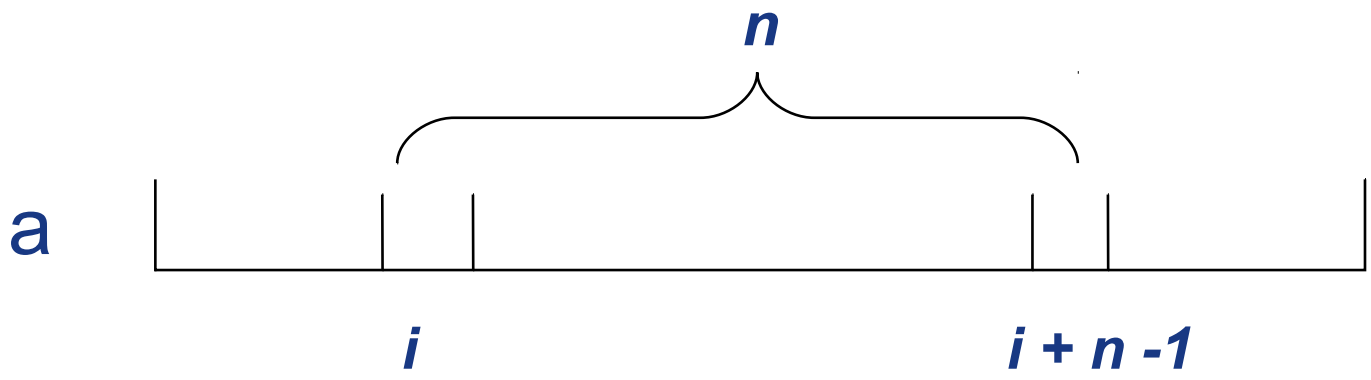
Para el caso base: La funcion de recurrencia insume un tiempo constate.

Para el caso recurrente: La funcion de recurrencia insume un tiempo constate mas el tiempo de invocar a la funcion con el parametro disminuido en uno.

Cálculo del Tiempo de Ejecución - Algoritmos Recursivos

Ejemplo :

Encontrar el máximo elemento en un arreglo de enteros tomando n posiciones a partir de la posición i .



Cálculo del Tiempo de Ejecución - Algoritmos Recursivos

```
#Calcula el Máximo en un arreglo.
 max(a, i, n )
        if (n == 1)
              return a[i]
        else
  m1 = max (a, i, n/2)
            m2 = max (a, i + (n/2), n/2)
          if (m1<m2)
             return m2
          else
             return m1
```

Cálculo del Tiempo de Ejecución - Función de recurrencia

Máximo en un arreglo

$$T(n) = \begin{cases} cte_1 & n = 1 \\ 2T(n/2) + cte_2 & n > 1 \end{cases}$$

Cálculo del T(n) para algoritmos recursivos.

Cálculo del Tiempo de Ejecución - Función de recurrencia

Sea la siguiente función de recurrencia:

$$T(n) = \begin{cases} 1 & n = 1 \\ 8T(n/2) + n^3 & n > 1 \end{cases}$$

Cálculo del Tiempo de Ejecución – Resolución

$$T(n) = \begin{cases} 1, n = 1 \\ 8T\left(\frac{n}{2}\right) + n^3, n \geq 2 \end{cases}$$

$$T(n) = 8T\left(\frac{n}{2}\right) + n^3, n \geq 2$$

$$T(n) = 8\left[8T\left(\frac{\frac{n}{2}}{2}\right) + \left(\frac{n}{2}\right)^3\right] + n^3, n \geq 4$$

Cada T(n) se debe reemplazar por la expresión en la definición

Cada ocurrencia de n se debe reemplazar por el nuevo valor

Cálculo del Tiempo de Ejecución – Resolución

$$T(n) = \begin{cases} 1, n = 1 \\ 8T\left(\frac{n}{2}\right) + n^3, n \geq 2 \end{cases}$$

$$T(n) = 8T\left(\frac{n}{2}\right) + n^3, n \geq 2$$

$$T(n) = 8\left[8T\left(\frac{\frac{n}{2}}{2}\right) + \left(\frac{n}{2}\right)^3 \right] + n^3, n \geq 4$$

$$T(n) = 8\left[8T\left(\frac{n}{4}\right) + \frac{n^3}{2^3} \right] + n^3, n \geq 4$$

Cálculo del Tiempo de Ejecución – Resolución

$$T(n) = 8^2 T\left(\frac{n}{4}\right) + 8\frac{n^3}{2^3} + n^3, n \geq 4$$

$$T(n) = 8^2 T\left(\frac{n}{2^2}\right) + n^3 + n^3, n \geq 4$$

$$T(n) = 8^2 T\left(\frac{n}{2^2}\right) + 2n^3, n \geq 2^2$$

$$T(n) = 8^i T\left(\frac{n}{2^i}\right) + in^3, n \geq 2^i$$

Cálculo del Tiempo de Ejecución – Resolución

$$T(n) = 8^i T\left(\frac{n}{2^i}\right) + in^3, n \geq 2^i$$

$$\frac{n}{2^i} = 1$$

$$n = 2^i$$

$$i = Log_2(n)$$

Cálculo del Tiempo de Ejecución – Resolución

$$T(n) = 8^{Log_2(n)} T\left(\frac{n}{2^{Log_2(n)}}\right) + Log_2(n)n^3$$

$$T(n) = n^{Log_2(8)} T\left(\frac{n}{n}\right) + Log_2(n)n^3$$

$$T(n) = n^3 T(1) + Log_2(n)n^3$$

$$T(n) = n^3 + Log_2(n)n^3$$