Correctitud de algoritmos recursivos

- Se demuestra por inducción en el tamaño del problema a resolver (por ejemplo, la cantidad de elementos en un arreglo o el número de bits en un entero).
- La base de la recursión es la base de la inducción.
- Se debe demostrar que las llamadas recursivas son en efecto subproblemas, es decir, que no se tiene una recursión infinita.
- El paso inductivo se hace suponiendo que las llamadas recursivas funcionan correctamente para demostrar que la llamada actual también funciona correctamente.

Números de Fibonacci recursivos

- $F_0 = 0$, $F_1 = 1$ y $F_n = F_{n-2} + F_{n-1}$ para $n \geq 2$.
- Considere la función fib(n) que regresa n si $n \leq 1$ y fib($n-2$) + fib($n-1$) en caso contrario.
- Demostremos por inducción que fib(n) regresa F_n.
- Para $n = 0$ la llamada fib(n) regresa $0 = F_0$.
- Para $n = 1$ la llamada fib(n) regresa $1 = F_1$.
- Suponga que $n \geq 2$ y que para toda $0 \leq m < n$ la llamada fib(m) regresa F_m.
- Entonces, la llamada fib(n) regresa

$$\text{fib}(n-2) + \text{fib}(n-1) = F_{n-2} + F_{n-1} = F_n$$

que es lo que queríamos demostrar.

Máximo recursivo

- Sea $A_1, \ldots, A_n$ un arreglo con n enteros y considere la función máximo(n) que regresa A_1 si $n \leq 1$ y máx(máximo($n-1$), A_n) en caso contrario.
- Demostremos por inducción que máximo(n) regresa máx($A_1, \ldots, A_n$).
- Para $n = 1$ la llamada máximo(n) regresa A_1.
- Suponga que $n \geq 1$ y que máximo(n) regresa máx($A_1, \ldots, A_n$).
- Entonces, la llamada máximo($n+1$) regresa

$$\text{máx}(\text{máximo}(n), A_{n+1})$$

es decir

$$\text{máx}(\text{máx}(A_1, \ldots, A_n), A_{n+1})$$

o bien

$$\text{máx}(A_1, \ldots, A_n, A_{n+1}).$$

Multiplicación recursiva

- Para toda x real definimos $\lfloor x \rfloor$ como el mayor entero que no excede x (la parte entera de x).

Función multiplica(y, z)

- Si $z = 0$ entonces regresa 0.
- Si z es impar
  - entonces regresa multiplica($2y, \lfloor z/2 \rfloor$) + y,
  - si no regresa multiplica($2y, \lfloor z/2 \rfloor$).

- Demostremos por inducción en z que para toda $y, z \geq 0$ la llamada multiplica(y, z) regresa yz.

Demostración

- Para $z = 0$ la llamada multiplica(y, z) regresa 0.
- Ahora suponga que $z \geq 0$ y que para toda $0 \leq q \leq z$ la llamada multiplica(y, q) regresa yq.
- ¿Qué es lo que regresa multiplica($y, z+1$)?
- Si $z+1$ es impar entonces multiplica($y, z+1$) regresa

$$\begin{aligned} \text{multiplica}(2y, \lfloor (z+1)/2 \rfloor) + y &= 2y\lfloor (z+1)/2 \rfloor + y \\ &= 2y(z/2) + y \\ &= y(z+1). \end{aligned}$$

- Si $z+1$ es par entonces multiplica($y, z+1$) regresa

$$\begin{aligned} \text{multiplica}(2y, \lfloor (z+1)/2 \rfloor) &= 2y\lfloor (z+1)/2 \rfloor \\ &= 2y(z+1)/2 \\ &= y(z+1). \end{aligned}$$

Derivación de relaciones de recurrencia

- Para derivar una relación de recurrencia para el tiempo de ejecución $T(n)$ de un algoritmo recursivo:
  1. Decida qué será el tamaño n del problema.
  2. Vea cuáles valores de n se usan como base para la recursión. Puede ser un solo valor n_0 o varios.
  3. Calcule $T(n_0)$. Normalmente bastará decir que es constante.
  4. El valor general de $T(n)$ será normalmente la suma de varios valores de $T(m)$ (las llamadas recursivas) más el trabajo adicional hecho.

Forma general de una relación de recurrencia

- En general

$$T(n) = \begin{cases} c & \text{si } n = n_0 \\ aT(f(n)) + g(n) & \text{en otro caso} \end{cases}$$

donde:

1. c es el tiempo de ejecución del caso base,
2. n_0 es el tamaño del caso base,
3. a es la cantidad de llamadas recursivas,
4. $f(n)$ es el tamaño de las llamadas recursivas y
5. $g(n)$ es el tiempo de ejecución de todo el procesamiento no incluido en las llamadas recursivas.

- ¿Cómo cambia esto si las llamadas recursivas son de diferentes tamaños?

Solución de relaciones de recurrencia

- Una técnica se llama sustitución repetida.
- Dada una relación de recurrencia para $T(n)$:
  1. Sustituya algunas veces hasta que vea un patrón.
  2. Escriba una fórmula en términos de n y la cantidad i de sustituciones.
  3. Escoja el valor de i para que todas las referencias a $T(i)$ sean referencias al caso base.
  4. Calcule la suma resultante.
- No funcionará siempre, pero en la práctica funciona frecuentemente.

Un teorema general

Teorema

Si n es una potencia de c entonces la solución a la recurrencia

$$T(n) = \begin{cases} d & \text{si } n = 1 \\ aT(n/c) + bn & \text{si } n > 1 \end{cases}$$

está dada por:

- $T(n) \in O(n)$ si $a < c$,
- $T(n) \in O(n \log n)$ si $a = c$ o
- $T(n) \in O(n^{\log_c a})$ si $a > c$.