Resolviendo Recurrencias

Mauro Jaskelioff

16/03/2015

Análisis de Algoritmos

- Queremos poder evaluar la performance de **algoritmos**.
- Para esto utilizamos
  - Notación Asintótica
  - Modelo de Costo basado en Lenguaje
  - Analizamos trabajo (W) y profundidad (S)
- Al plantear el trabajo y profundidad de algoritmos recursivos surgen recurrencias.
- ¿Cómo resolver las recurrencias?
  - Veremos varias técnicas...

Substitución: Adivinando la solución

- Probamos los siguientes pasos
  1. Adivinamos la forma de la solución.
  2. Probamos que es correcta usando inducción matemática.
- Ejemplo (recordar mergesort)

$$
\begin{aligned}
W(0) &= c_0 \\
W(1) &= c_1 \\
W(n) &= 2W(\lfloor n/2 \rfloor) + c_2 n
\end{aligned}
$$

- Adivinamos que $W(n) \in O(n \lg n)$
- Probamos que $W(n) \leqslant cn \lg n$.

- Reemplazamos nuestra solución en la recurrencia

$$\begin{aligned} W(n) &\leqslant 2(c\lfloor n/2\rfloor \lg\lfloor n/2\rfloor) + c_2 n \\ &\leqslant cn\lg(n/2) + c_2 n \\ &= cn\lg n - cn\lg 2 + c_2 n \\ &= cn\lg n - cn + c_2 n \\ &\leqslant cn\lg n \qquad\qquad \text{siempre y cuando } c \geqslant c_2. \end{aligned}$$

- Faltan chequear los casos bases. ¿$W(0) \leqslant c \cdot 0 \cdot \lg 0$? ¿$W(1) \leqslant c \cdot 1 \cdot \lg 1$?
- La notación O sólo requiere $W(n) \leqslant cn\lg n$ para $n \geqslant N$. Tomamos $N = 2$

$$\begin{aligned} W(2) &= 2W(1) + 2c_2 = 2c_1 + 2c_2 \leqslant c \cdot 2\lg 2 = 2c \\ W(3) &= 2W(1) + 3c_2 = 2c_1 + 3c_2 \leqslant c \cdot 3\lg 3 \end{aligned}$$

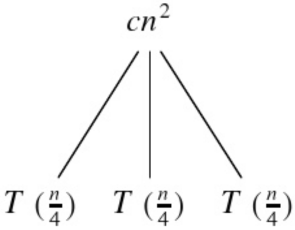
- Elegimos c suficientemente grande.

Árboles de recurrencia

- Una técnica general para resolver recurrencias son los *árboles de recurrencia*
- Los ilustramos usando la recurrencia

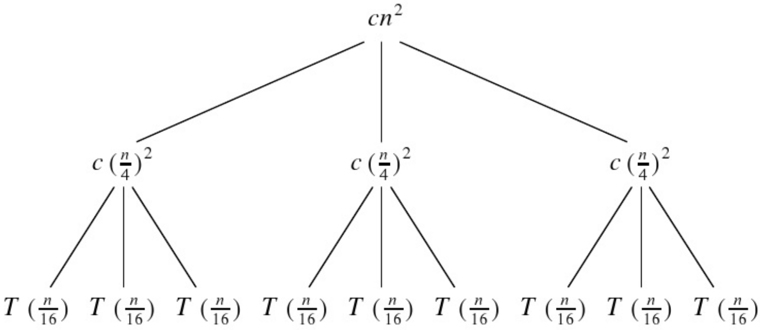
$$T(1) = c_1$$
$$T(n) = 3T(n/4) + cn^2$$

- Hacemos un árbol que en su raíz tiene el costo para el n inicial, con ramas indicando cada una de las llamadas recursivas.



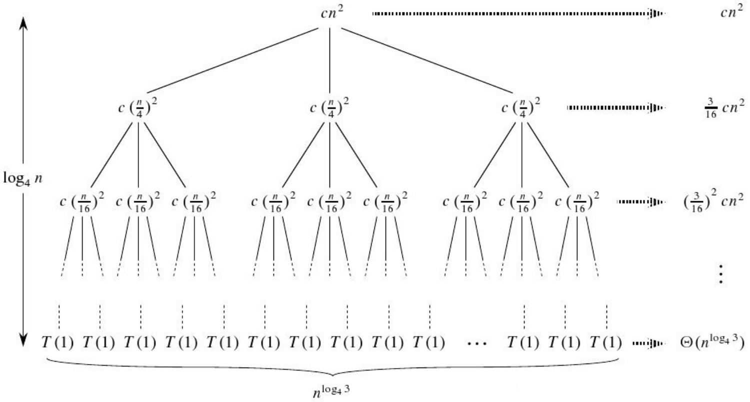
Árboles de recurrencia (cont.)

- Las ramas correspondientes a las llamadas recursivas se expanden, ahora tienen el costo correspondiente a cada llamada y generan nuevas hojas.



Árboles de recurrencia (cont.)

- Se sigue expandiendo el árbol hasta llegar a un caso base.
- En cada nivel se suma el total de operaciones por nivel.
  El costo total es la suma de todos los niveles.



Árboles de recurrencia (cont.)

- La suma obtenida se manipula algebraicamente para llegar al resultado.

$$
\begin{aligned}
T(n) &= cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \cdots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + kn^{\log_4 3} \\
&= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + kn^{\log_4 3} \\
&= \frac{(3/16)^{\log_4 n} - 1}{(3/16) - 1} cn^2 + kn^{\log_4 3} \\
&\in O(n^2)
\end{aligned}
$$

- Si somos prolijos, el método nos da una solución exacta.
- Si no, al menos nos da un candidato para usar el método de substitución.

Ejercicio

Verificar usando el método de la substitución que la cota obtenida es ajustada. Es decir, probar que

$$T(n) = 3T(n/4) + cn^2 \quad \in \Theta(n^2)$$

Nota: De aquí en más, si no se menciona caso base,
suponerlo(s) constante(s).

Simplificando Recurrencias

- Para el análisis de mergesort, nos olvidamos de los $\lfloor \cdot \rfloor$ y los $\lceil \cdot \rceil$
- ¿Cuando es válido hacer estas aproximaciones?
- Si $n = b^k$ entonces $n/b = \lfloor n/b \rfloor = \lceil n/b \rceil$.
- Notar que no alcanza que n sea múltiplo de b.
- Podría suceder que la solución asintótica para entradas $n = b^k$ sea la solución para cualquier entrada.

Funciones suaves

- Una función $f : \mathbb{N} \to \mathbb{R}^+$ es *eventualmente no decreciente* si

$$\exists\, N \in \mathbb{N}.\ f(n) \leqslant f(n+1) \text{ para todo } n \geqslant N$$

- Una función $f : \mathbb{N} \to \mathbb{R}^+$ es *b-suave* (smooth) si
    1. es eventualmente no decreciente y
    2. $f(bn) \in O(f(n))$.
- Una función es *suave* si es b-suave para todo b.
- Propiedad: si f es b-suave para un $b \geqslant 2$, entonces es suave para todo b.
- Ejemplos de funciones suaves: n^2, n^r (para todo r), $n \lg n$.
- Ejemplos de funciones no suaves: $n^{\lg n}$, 2^n, $n!$.
- Ejercicio: verificar que n^2 es suave y que 2^n no lo es.

Regla de suavidad

- *Regla de suavidad:* Sea f suave y sea g eventualmente no decreciente. Para todo $b \geqslant 2$,

$$g(b^k) \in \Theta(f(b^k)) \quad \Rightarrow \quad g(n) \in \Theta(f(n))$$

- Ejemplo de uso: Considere la siguiente recurrencia

$$\begin{aligned} W(1) &= c \\ W(n) &= W(\lceil n/2 \rceil) + W(\lfloor n/2 \rfloor) + kn \end{aligned}$$

- Si sólo consideramos potencias de 2 obtenemos

$$\begin{aligned} W'(1) &= c \\ W'(n) &= 2W'(n/2) + kn \end{aligned}$$

que es $O(n \lg n)$. Como $n \lg n$ es suave, $W(n) \in O(n \lg n)$.

Ecuación Característica

- Método sistemático.
- Permite resolver recurrencias de forma exacta.
- Limitado a ciertas formas de recurrencias.
- Muy potente.

Recurrencias Lineales Homogéneas

- Las *recurrencias lineales homogéneas a coeficientes constantes* son de la forma:

$$a_0 T(n) + a_1 T(n-1) + \cdots + a_k T(n-k) = 0$$

  - Son lineales ya que no contienen términos como $T(n-i)T(n-j)$, $T^2(n-1)$.
  - Son homogéneas porque la combinación lineal de los $T(n-i)$ es igual a 0.
  - A coeficientes constantes porque los a_i son constantes.

- Ejemplo: $F(n) = F(n-1) + F(n-2)$.
  Se puede reescribir como

$$F(n) - F(n-1) - F(n-2) = 0 \qquad k = 2, a_0 = 1, a1 = -1, a_2 = -1$$

Solución general

- Dada la recurrencia

$$a_0 T(n) + a_1 T(n-1) + \cdots + a_k T(n-k) = 0$$

definimos su polinomio característico como

$$p(x) = a_0 x^k + a_1 x^{k-1} + \cdots + a_k$$

- Toda solución de será de la forma

$$T(n) = \sum_{i=1}^{k} c_i r_i^n$$

donde los r_i son las raíces del polinomio, bajo el supuesto que todas las r_i son distintas (no hay raíces múltiples).

- Las constantes $c_1, c_2, \ldots, c_k$ se obtienen de k condiciones iniciales.

Ejemplo : Fibonacci

$$F(0) = 0$$
$$F(1) = 1$$
$$F(n) = F(n-1) + F(n-2)$$

- La recurrencia es $F(n) - F(n-1) - F(n-2) = 0$ y su polinomio característico es $x^2 - x - 1$.
- Las raíces son $r_1 = \frac{1+\sqrt{5}}{2}$ y $r_2 = \frac{1-\sqrt{5}}{2}$
- La solución general es $F(n) = c_1 r_1^n + c_2 r_2^n$.
- $0 = F(0) = c_1 + c_2$ $\qquad 1 = F(1) = r_1 c_1 + r_2 c_2$

$$\begin{array}{ccccc} c_1 & + & c_2 & = 0 \\ r_1 c_1 & + & r_2 c_2 & = 1 \end{array} \quad \Rightarrow \quad \begin{cases} c_1 & = \frac{1}{\sqrt{5}} \\ c_2 & = -\frac{1}{\sqrt{5}} \end{cases}$$

- Por lo tanto $F(n) = \frac{1}{\sqrt{5}} \left[\left(\frac{1+\sqrt{5}}{2} \right)^n - \left(\frac{1-\sqrt{5}}{2} \right)^n \right]$ (Moivre)

Ejercicio

Resolver la recurrencia

$$\begin{aligned}
&W\ (0) = 0\\
&W\ (1) = 5\\
&W\ (n) = 3\ W\ (n-1) + 4\ W\ (n-2)
\end{aligned}$$

- $W(n) - 3W(n-1) - 4W(n-2) = 0$
- El polinomio característico es $x^2 - 3x - 4$ $\quad r_1 = -1 \quad r_2 = 4$
- La solución es de la forma $W(n) = c_1(-1)^n + c_2 4^n$
- El sistema de ecuaciones dado por las condicionales iniciales es

$$\begin{array}{rcrl}
c_1 & + & c_2 & = 0\\
-c_1 & + & 4c_2 & = 5
\end{array}$$

- Resolviendo, obtenemos $c_1 = -1$ y $c_2 = 1$.
- Por lo tanto $W(n) = 4^n - (-1)^n$

Raíces Múltiples

- ¿Qué sucede cuando hay raíces múltiples?
- El polinomio característico $p(x) = a_0 x^k + a_1 x^{k-1} + \cdots + a_k$ tendrá
  - q raíces distintas $r_1, r_2, \ldots, r_q$.
  - cada una con multiplicidad $m_1, m_2, \ldots, m_q$.
- Las soluciones serán de la forma

$$T(n) = \sum_{i=1}^{q} \sum_{j=0}^{m_i - 1} c_{ij} n^j r_i^n$$

- Las constantes c_{ij} se determinan con las condiciones iniciales.

Ejemplo

$$S(0) = 0$$
$$S(1) = 1$$
$$S(2) = 2$$
$$S(n) = 5\,S(n-1) - 8\,S(n-2) + 4\,S(n-3)$$

- Reescribimos la recurrencia como $S(n) - 5S(n-1) + 8S(n-2) - 4S(n-3) = 0$
- El polinomio característico es $x^3 - 5x^2 + 8x - 4 = (x-1)(x-2)^2$
- Las raíces son $r_1 = 1$ con multiplicidad 1 y $r_2 = 2$ con multiplicidad 2
- La solución general es $S(n) = c_1 1^n + c_2 2^n + c_3 n 2^n$

Ejemplo (cont.)

- La solución general es $S(n) = c_1 1^n + c_2 2^n + c_3 n 2^n$
- Las condiciones iniciales nos dan

$$\begin{array}{ccccccl} c_1 & + & c_2 & & & = 0 & \quad n = 0 \\ c_1 & + & 2c_2 & + & 2c_3 & = 1 & \quad n = 1 \\ c_1 & + & 4c_2 & + & 8c_3 & = 2 & \quad n = 2 \end{array}$$

- $c_1 = -2,\ c_2 = 2,\ c_3 = -\frac{1}{2}$
- La solución queda

$$S(n) = 2^{n+1} - n2^{n-1} - 2$$

Recurrencias Lineales No Homogéneas

- La solución de recurrencias lineales se complica cuando la recurrencia no es homogénea.
- Consideraremos recurrencias no homogéneas con la siguiente forma

$$a_0 T(n) + a_1 T(n-1) + \cdots + a_k T(n-k) = b_1^n p_1(n) + b_2^n p_2(n) + \ldots$$

  - Las b_i son constantes **distintas**.
  - Cada p_i es un polinomio de grado d_i.
- Estas recurrencias se resuelven usando el siguiente polinomio característico

$$(a_0 x^k + a_1 x^{k-1} + \cdots + a_k)(x - b_1)^{d_1+1}(x - b_2)^{d_2+1} + \ldots$$

- Luego, procedemos como en el caso homogéneo.

Ejemplo

$$T\ (0) = 0$$
$$T\ (n) = 2\ T\ (n-1) + n + 2^n$$

- La recurrencia se puede reescribir como

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

que con $b_1 = 1$, $p_1(n) = n$, $b_2 = 2$, $p_2(n) = 1$ tiene la forma adecuada.

- $d_1 = 1$ y $d_2 = 0$, por lo que el polinomio característico es

$$(x-2)(x-1)^2(x-2)$$

$r_1 = 1$ y $r_2 = 2$, ambas con multiplicidad 2.

- Por lo tanto las soluciones serán de la forma

$$T(n) = c_1 1^n + c_2 n 1^n + c_3 2^n + c_4 n 2^n$$

Ejercicio

$$T\ (0) = 1$$
$$T\ (n) = 3\ T\ (n-1) + 2^n$$

- Reescribimos la recurrencia como $T(n) - 3T(n-1) = 2^n$. $(b_1 = 2,\ p_1(n) = 1,\ d_1 = 0)$
- El polinomio característico es $(x-3)(x-2)$
- Las raíces son 3 y 2 con multiplicidad 1.
- Las soluciones son de la forma $T(n) = c_1 3^n + c_2 2^n$.
- Determinamos las constantes $c_1 = 3$ y $c_2 = -2$.
- La solución final es $T(n) = 3^{n+1} - 2^{n+1}$.

Cambio de Variable

- Dada la recurrencia

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

- Definimos una nueva recurrencia $T'(i) = T(2^i)$.
- O sea

$$T'(i) = \begin{cases} 1 & \text{si } i = 0 \\ 3T'(i-1) + 2^i & \text{si } i \geqslant 1 \end{cases}$$

- $T'(i) = 3^{i+1} + 2^{i+1}$.
- $T'(i) = T(2^i) \quad \Leftrightarrow \quad T(n) = T'(\lg n)$ para n potencia de 2!
- Si n es potencia de 2, $T(n) = T'(\lg n) = 3 \cdot n^{\lg 3} - 2n$

$$T(n) \in \Theta(n^{\lg 3}) \text{ si } n \text{ potencia de } 2$$

- Pero $n^{\lg 3}$ es suave y $T(n)$ creciente $\Rightarrow T(n) \in \Theta(n^{\lg 3})$

Ejercicio

$$\begin{aligned} T(0) &= k \\ T(n) &= 2T(n/2) + n \lg n \end{aligned}$$

- $T'(i) = T(2^i) = 2T(2^{i-1}) + i2^i = 2T'(i-1) + i2^i$
- $T'(i) - 2T'(i-1) = i2^i$, con pol. car.
  $(x-2)(x-2)^2 = (x-2)^3$ (raíz 2 con multiplicidad 3)
- $T'(i) = c_1 2^i + c_2 i 2^i + c_3 i^2 2^i$
- $T(n) = c_1 n + c_2 n \lg n + c_3 n \lg^2 n$ (para n potencia de 2)
- $n \lg n = (c_2 - c_3)n + 2c_3 n \lg n \quad \Rightarrow \quad c_2 = c_3 = \frac{1}{2}$
- $T(n) \in \Theta(n \lg^2 n)$ para n potencia de 2.
- $n \lg^2 n$ suave y $T(n)$ creciente $\Rightarrow$ $T(n) \in \Theta(n \lg^2 n)$

Resolviendo recurrencias

- Las recurrencias que aparecen en algoritmos "Divide & Conquer" usualmente tienen la forma:

$$T(n) = aT(n/b) + f(n)$$

- En general, podemos obtener el orden de estas recurrencias directamente usando una resolución general.

"El Teorema Maestro"

El Teorema Maestro

- Dados $a \geqslant 1$ y $b > 1$ y la recurrencia

$$T(n) = aT(n/b) + f(n) \ ,$$

entonces

$$T(n) \in \begin{cases} \Theta(n^{\lg_b a}) & \text{si } \exists \epsilon > 0.\ f(n) \in O(n^{\lg_b a - \epsilon}) \\ \Theta(n^{\lg_b a} \lg n) & \text{si } f(n) \in \Theta(n^{\lg_b a}) \\ \Theta(f(n)) & \begin{array}{l} \text{si } \exists \epsilon > 0.\ f(n) = \Omega(n^{\lg_b a + \epsilon}) \\ \text{y } \exists c < 1, N \in \mathbb{N}.\ \forall n > N. \\ af(n/b) \leqslant cf(n) \end{array} \end{cases}$$

- Para $W(n) = 2W(\lfloor n/2 \rfloor) + c_2 n$ estamos en el 2do caso ya que $\lg_2 2 = 1$ y $g(n) \in \Theta(n^1)$, y por lo tanto $W(n) = \Theta(n \lg n)$

Comentarios acerca del Teorema Maestro

- Los casos se deciden comparando $f(n)$ y $n^{\lg_b a}$.
  - Caso 1: $n^{\lg_b a}$ es mas grande.
  - Caso 2: $f(n)$ y $n^{\lg_b a}$ tienen el mismo orden.
  - Caso 3: $f(n)$ es mas grande.
- En realidad no basta con que una sea mas grande que la otra, sino que debe serlo polinomialmente.
  - $f(n) = n \lg n$ no es polinomialmente mas grande que $g(n) = n$.
- Los tres casos **no** cubren todas las posibilidades
- En la práctica van a probar una versión (levemente) menos general del teorema maestro.

Resumen

Resolución de recurrencias usando:

- Substitución
- Árbol de recurrencias.
- Regla de suavidad
- Ecuación característica
- Teorema Maestro

Referencias

- Introduction to Algorithms. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein
- Fundamentals of Algorithmics. Gilles Brassard, Paul Bratley