Algoritmi e strutture dati

Soluzione di ricorrenze

Menú di questa lezione

In questa lezione vedremo tre metodi classici di soluzione di ricorrenze. Introdurremo in maniera formale il Master Theorem e ne vedremo la dimostrazione.

Soluzione di ricorrenze

Con le nozioni viste finora, per ogni $T(n)$ **esplicita** (come ad esempio quella di *InsertionSort*) possiamo trovare il suo ordine di grandezza (quindi, ad esempio, possiamo dire che *InsertionSort* ha complessità, nel caso peggiore, $\Theta(n^2)$). Ma ancora non sappiamo esplicitare una ricorrenza. Esistono vari metodi per trasformare una ricorrenza da **implicita** a **esplicita**, cioè, per risolverla. L'esempio che dobbiamo ancora risolvere é quello di *RecursiveBinarySearch*, che, abbiamo detto, ha complessità, nel caso peggiore, $T(n) = T(\frac{n}{2}) + 1$. Sui testi, troviamo che esistono tre modi classici per la soluzione delle ricorrenze: il metodo dell'**albero di ricorsione** (o sviluppo in serie), il cosiddetto **Master Theorem** per le ricorrenze, e il metodo della **sostituzione** (o induzione). Invece di studiare questi metodi separatamente, cerchiamo di comprendere in profondità cosa significa sviluppare una ricorrenza.

Soluzione di ricorrenze: sviluppo

Il metodo dello sviluppo si basa sull'idea di, appunto, **sviluppare** la ricorrenza per cercare di estrarne il comportamento. Infatti, se ad esempio $T(n) = T(\text{frazione di } n) + f(n)$, allora per calcolare $T(\text{frazione di } n)$ posso ri-applicare la stessa ricorrenza, e ottenere una forma, ancora implicita, ma con un termine in più. Vediamo un esempio concreto.

Soluzione di ricorrenze: sviluppo

Supponiamo di voler risolvere proprio $T(n) = T(\frac{n}{2}) + 1$.

Da $T(n) = T(\frac{n}{2}) + 1$, abbiamo che:

$$\begin{array}{lll} T(n) & = T(\frac{n}{2}) + 1 & \text{ricorrenza} \\ & = (T(\frac{n}{4}) + 1) + 1 & = T(\frac{n}{4}) + 2 \\ & = (T(\frac{n}{8}) + 1) + 2 & = T(\frac{n}{8}) + 3 \\ \ldots, & & \end{array}$$

finché é possibile dividere, cioé finchè l'argomento della T diventa 1 o meno (e quindi non si puó piú dividere). Assumendo che $T(1) = 1$, si ottiene che $T(n) = \lfloor log(n) \rfloor + 1 = \Theta(log(n))$. Questa è una forma molto semplice di **albero di ricorsione**, che ha un solo ramo.

$$T(n) = T\left(\frac{n}{2}\right) + 1 \qquad I_P:\ T(1) = 1$$

Bd!

$n = 8$

4

2

1

$$\begin{array}{c} T(n) \\ | \\ T\left(\frac{n}{2}\right) \\ | \\ T\left(\frac{n}{4}\right) \\ | \\ T\left(\frac{n}{8}\right) \\ \vdots \\ \hline T(1) \end{array} \qquad \left.\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ \vdots \end{array}\right\} \lg(n)$$

$$\Rightarrow T(n) = \lg(n) \cdot 1 + 1 \overset{!}{=} \lg(n) = \Theta(\lg(n))$$

Soluzione di ricorrenze: sviluppo

Consideriamo, come altro esempio, la ricorrenza $T(n) = 2 \cdot T(\frac{n}{2}) + n$.
Sviluppando, otteniamo:

$$\begin{array}{lll} T(n) & = 2 \cdot T(\frac{n}{2}) + n & \text{ricorrenza} \\ & = 2 \cdot (2 \cdot T(\frac{n}{4}) + \frac{n}{2}) + n & = 4 \cdot T(\frac{n}{4}) + 2 \cdot n \\ & = 4 \cdot (2 \cdot T(\frac{n}{8}) + \frac{n}{4}) + 2 \cdot n & = 8 \cdot T(\frac{n}{8}) + 3 \cdot n \\ \ldots, & & \end{array}$$

finchè è possibile dividere, cioè finchè l'argomento diventa 1 o meno.
Assumendo che $T(1) = 1$, otteniamo $T(n) = 2^{log(n)} + n \cdot log(n)$, cioè $T(n) = n + n \cdot log(n)$, cioè $T(n) = \Theta(n \cdot log(n))$.

$$T(n) = 2T\left(\frac{n}{2}\right) + n \qquad (2n,\ n+5,\ cn,\ 3n+4n\ \dots)$$

$$\Theta(n)$$

$$T(n) \rightsquigarrow n$$

$$T\left(\frac{n}{2}\right)\ T\left(\frac{n}{2}\right) \rightsquigarrow \frac{n}{2} + \frac{n}{2} = n$$

$$T\left(\frac{n}{4}\right)\ T\left(\frac{n}{4}\right)\ T\left(\frac{n}{4}\right)\ T\left(\frac{n}{4}\right) \rightsquigarrow \frac{n}{4} \cdot 4 = n$$

$$\vdots$$

$$\boxed{T(1)\ T(1)\ \cdots\cdots\ T(1)}$$

$$\lim_{n \to \infty} \frac{n}{n\lg(n)} \to 0$$

$$\boxed{\sum_{i=0}^{\lg(n)-1} n} = n\lg(n)$$

$$\Theta(n\lg(n))$$

$$2^{\lg(n)}\, T(1) = n \cdot 1 = \boxed{n} \;\Rightarrow\; T(n) = n\lg(n) + \cancel{n}$$

Soluzione di ricorrenze: sviluppo

Sviluppiamo adesso, come ultimo esempio, la ricorrenza $T(n) = 3 \cdot T(\frac{n}{2}) + n^2$. Otteniamo:

$$
\begin{array}{lll}
T(n) & = 3 \cdot T(\frac{n}{2}) + n^2 & \textit{ricorrenza} \\
 & = 3 \cdot (3 \cdot T(\frac{n}{4}) + (\frac{n}{2})^2) + n^2 & = 9 \cdot T(\frac{n}{4}) + \frac{3}{4} \cdot n^2 + n^2 \\
 & = 9 \cdot (3 \cdot T(\frac{n}{8}) + (\frac{n}{4})^2) + \frac{3}{4} \cdot n^2 + n^2 & = 27 \cdot T(\frac{n}{8}) + \ldots \\
\ldots & &
\end{array}
$$

arrivando alla seguente forma, sempre sotto l'ipotesi che $T(1) = 1$:

$$3^{log(n)} + \Sigma_{i=0}^{log(n)-1} (\tfrac{3}{4})^i \cdot n^2 = n^{log(3)} + n^2 \cdot \Sigma_{i=0}^{log(n)-1} (\tfrac{3}{4})^i = \Theta(n^2),$$

ricordando che:

$$x^{log(y)} = (2^{log(x)})^{log(y)} = 2^{log(y) \cdot log(x)} = (2^{log(y)})^{log(x)} = y^{log(x)}.$$

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

$T(n)$ ⟿ n^2

$T\left(\frac{n}{2}\right)$ $T\left(\frac{n}{2}\right)$ $T\left(\frac{n}{2}\right)$ ⟿ $\frac{3}{4}n^2$ ⟿ $\left(\frac{3}{4}\right)^i n^2$

$T\left(\frac{n}{4}\right)$ $T\left(\frac{n}{4}\right)$ $T\left(\frac{n}{4}\right)$ ⟿ $\frac{9}{16}n^2$

$$\sum_{i=0}^{\lg(n)-1} \left(\frac{3}{4}\right)^i n^2$$

$T(1)$ . . . . $T(1)$

$$T(n) = 3^{\lg(n)} + n^2 \underbrace{\sum_{i=0}^{\lg(n)-1} \left(\frac{3}{4}\right)^i}$$

SOMMATORIA GEOMETRICA

(CONTINUA)

$$T(n) = 3^{\lg(n)} + n^2 \sum_{i=0}^{\log(n)-1} \left(\frac{3}{4}\right)^i$$

$0 < 2 < 1$

Siccome la serie corrispondente converge la posso eliminare

$$\sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^i = c$$

$$T(n) = 3^{\lg(n)} + n^2 \cdot c$$

$$= n^{\lg_2(3)} + n^2 \cdot c$$

$$= n^{1,\dots} + n^2 \cdot c = \Theta(n^2)$$

$$3^{\lg_2(n)} = \left(2^{\lg_2(3)}\right)^{\lg_2(n)} =$$

$$2^{\lg_2(3)\cdot \lg_2(n)} = 2^{\lg_2(n)\cdot \lg_2(3)} =$$

$$\left(2^{\lg_2(n)}\right)^{\lg_2(3)} = n^{\lg_2(3)}$$

Soluzione di ricorrenze: il Master Theorem

Generalizzando quanto visto, ci accorgiamo che esiste una certa **forma** di ricorrenza tale che, se la ricorrenza ha quella forma, la sua soluzione **sembra** abbastanza semplice. Questa forma, in generale, è:

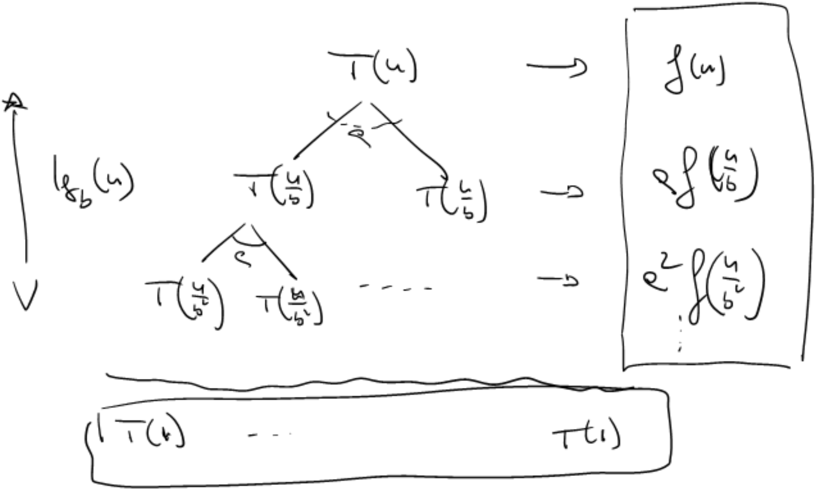
$$T(n) = a \cdot T(\frac{n}{b}) + f(n)$$

Infatti, le ricorrenze in questa forma si possono facilmente sviluppare e/o testare per sostituzione. Possiamo generalizzare questo concetto ed evitare, così, di ripetere molti passi? La prima osservazione è che se sviluppiamo **in modo generale** questa ricorrenza otteniamo:

$$T(n) = \Sigma_{i=0}^{log_b(n)-1} a^i \cdot f(\frac{n}{b^i}) + \Theta(n^{log_b(a)})$$

sotto l'ipotesi che n sia una **potenza esatta** di b. I ragionamenti che seguono possono essere ulteriormente generalizzati quando questa ipotesi non è rispettata.

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$



$$T(n) = \sum_{i=0}^{\log_b(n)-1} a^i f\left(\frac{n}{b^i}\right) + a^{\log_b(n)} = \sum_{i=0}^{\log_b(n)-1} a^i f\left(\frac{n}{b^i}\right) + n^{\log_b(a)}$$

Soluzione di ricorrenze: il Master Theorem

Caso I del M.T.

Supponiamo adesso che $f(n)$ sia di ordine **polinomicamente** inferiore a $n^{log_b(a)}$; questo si scriverebbe: $f(n) = O(n^{log_b(a)-\epsilon})$ per un certo $\epsilon > 0$. In questo caso, valutiamo la prima parte dell'espressione precedente:

$$
\begin{aligned}
\Sigma_{i=0}^{log_b(n)-1} a^i \cdot f(\tfrac{n}{b^i}) &= O(\Sigma_{i=0}^{log_b(n)-1} a^i \cdot (\tfrac{n}{b^i})^{log_b(a)-\epsilon}) && \textit{sostituzione} \\
&= O(\Sigma_{i=0}^{log_b(n)-1} \tfrac{a^i \cdot n^{log_b(a)-\epsilon}}{b^{i\cdot(log_b(a)-\epsilon)}}) && \textit{moltiplicazione} \\
&= O(n^{log_b(a)-\epsilon} \cdot \Sigma_{i=0}^{log_b(n)-1} \tfrac{a^i \cdot b^{i\cdot\epsilon}}{b^{i\cdot log_b(a)}}) && \textit{estrazione} \\
&= O(n^{log_b(a)-\epsilon} \cdot \Sigma_{i=0}^{log_b(n)-1} (\tfrac{a\cdot b^{\epsilon}}{b^{log_b(a)}})^i) && \textit{esponente} \\
&= O(n^{log_b(a)-\epsilon} \cdot \Sigma_{i=0}^{log_b(n)-1} (b^{\epsilon})^i) && \textit{prop. logaritmi}
\end{aligned}
$$

Ci accorgiamo che la sommatoria è nota: si tratta di una sommatoria geometrica, che sappiamo sempre valutare.

$\Rightarrow$ es: 4^4 vs 4^3

$4^{3.2}$ vs 4^3 ...

$$\frac{a^i \cdot n^{\left(\log_b(a) - \varepsilon\right)}}{b^{i\left(\log_b(a) - \varepsilon\right)}} = \frac{a^i\, n^{\log_b(a) - \varepsilon}}{\left(\frac{b^{i \log_b(a)}}{b^{i\varepsilon}}\right)} = n^{\log_b(a) - \varepsilon} \cdot \frac{a^i\, b^{i\varepsilon}}{b^{i \log_b(a)}}$$

$$= \frac{\cancel{a^i} \cdot b^{\varepsilon i}}{\cancel{a^i}} = \left(b^{\varepsilon}\right)^i$$

$$\boxed{\sum_{i=0}^{n} x^i = \frac{1-x^{n+1}}{1-x} = \frac{x^{n-1}-1}{x-1}}$$

Som. geometrica

Quindi:

$$\begin{aligned}
\Sigma_{i=0}^{\log_b(n)-1} a^i \cdot f(\tfrac{n}{b^i}) &= O(n^{\log_b(a)-\epsilon} \cdot \tfrac{b^{\epsilon \cdot \log_b(n)}-1}{b^\epsilon - 1}) && \textit{somma geometrica} \\
&= O(n^{\log_b(a)-\epsilon} \cdot \tfrac{n^\epsilon - 1}{b^\epsilon - 1}) && \textit{prop. logaritmi} \\
&= O(n^{\log_b(a)}) && \textit{moltip. e def. di } O()
\end{aligned}$$

Mettendo tutto nell'espressione dalla quale siamo partiti:

$$T(n) = O(n^{\log_b(a)}) + \Theta(n^{\log_b(a)}) = \Theta(n^{\log_b(a)}).$$

$$\frac{\boxed{n^{\log_b(a)-\epsilon} \cdot n^\epsilon} - \cancel{n^{\log_b(a)-\epsilon}}}{\cancel{b^\epsilon - 1}}$$

Caso II M.T.

Cosa succederebbe, invece, se $f(n) = \Theta(n^{log_b(a)})$? Avremmo che

$$
\begin{array}{rcl}
\Sigma_{i=0}^{log_b(n)-1} a^i \cdot f(\frac{n}{b^i}) & = & \Theta(\Sigma_{i=0}^{log_b(n)-1} a^i \cdot (\frac{n}{b^i})^{log_b(a)}) \\
& = & \Theta(n^{log_b(a)} \cdot \Sigma_{i=0}^{log_b(n)-1} \frac{a^i}{b^{log_b(a)}}) \quad = \Sigma \frac{\cancel{a^i}}{\cancel{a^i}} \\
& = & \Theta(n^{log_b(a)} \cdot \Sigma_{i=0}^{log_b(n)-1} 1) \\
& = & \Theta(n^{log_b(a)} \cdot log_b(n))
\end{array}
$$

E di nuovo, mettendo tutto nell'espressione dalla quale siamo partiti:

$$T(n) = \Theta(n^{log_b(a)} \cdot log_b(n)) + \Theta(n^{log_b(a)}) = \Theta(n^{log_b(a)} \cdot log_b(n)).$$

$$\lim_{n \to \infty} \frac{\cancel{n^{log_b(a)}}}{\cancel{n^{log_b(a)}} \cdot log_b(n)} = \lim_{n \to \infty} \frac{1}{log_b(n)} \to 0$$

 Caso III n.n.

Supponiamo, infine, che $\boxed{f(n) = \Omega(n^{\log_b(a)+\epsilon})}$ (cioè $f(n)$ polinomicamente di grado superiore a $n^{\log_b(a)}$), che implica $f(n) = \Omega(n^{\log_b(a)})$. Immaginiamo anche che $\boxed{a \cdot f(\frac{n}{b}) \leq c \cdot f(n)}$ per qualche $c < 1$ e tutti gli $n \geq n_0$. Allora:

$$
\begin{array}{rcll}
\Sigma_{i=0}^{\log_b(n)-1} a^i \cdot f(\frac{n}{b^i}) & \leq & \Sigma_{i=0}^{\log_b(n)-1} c^i \cdot f(n) & \\
& \leq & f(n) \cdot \Sigma_{i=0}^{\infty} c^i & \textit{maggiorazione} \\
& = & f(n) \cdot \frac{1}{1-c} & \textit{somma di serie geom.}
\end{array}
$$

Passo da sommatoria a serie

Pertanto:

$$\Sigma_{i=0}^{\log_b(n)-1} a^i \cdot f(\frac{n}{b^i}) = \Theta(f(n))$$

Ma questo implica:

$\Theta(f(n) + n^{\log_b(a)})$

$$T(n) = \Theta(f(n)) + \Theta(n^{\log_b(a)}) = \Theta(f(n))$$

considerato che $f(n)$ è polinomicamente di grado superiore a $n^{\log_b(a)}$, di nuovo, per ipotesi.

Soluzione di ricorrenze: il Master Theorem

Se mettiamo tutto assieme, generalizzando per n **non necessariamente** potenza perfetta di b, e intendendo $\frac{n}{b}$ come $\lfloor \frac{n}{b} \rfloor$ oppure $\lceil \frac{n}{b} \rceil$), indifferentemente, otteniamo che se:

$$T(n) = a \cdot T(\frac{n}{b}) + f(n)$$

allora questa può essere risolta attraverso l'applicazione di quello che è diventato noto come il **Master Theorem** per le ricorrenze, introdotto da Jon Bentley, Dorothea Haken, and James Saxe nel 1980.

Master Theorem generalizzato

$$T(n) = \begin{cases} \Theta(n^{log_b(a)}) & \text{se } f(n) = O(n^{log_b(a)-\epsilon}) \quad \text{caso I} \\ & \text{per qualche } \epsilon > 0 \\ \Theta(n^{log_b(a)} \cdot log^{k+1}(n)) & \text{se } f(n) = \Theta(n^{log_b(a)} \cdot log^{k}(n)) \quad \text{caso II} \\ & \text{per qualche } k \geq 0 \\ \Theta(f(n)) & \text{se } f(n) = \Omega(n^{log_b(a)+\epsilon}) \quad \text{caso III} \\ & \text{e se esiste } c < 1 \text{ t.c. } a \cdot f(\frac{n}{b}) \leq c \cdot f(n) \\ & \text{per ogni } n \geq n_0 \end{cases}$$

Soluzione di ricorrenze: il Master Theorem

$T(n) = a\,T\left(\frac{n}{b}\right) + f(n)$

Usiamo il teorema per la ricorrenza, giá vista, $T(n) = 2 \cdot T(\frac{n}{2}) + n$. Abbiamo che il caso corretto è il numero 2, con $a = b = 2$, $k = 0$, $f(n) = n$, per ottenere, come è lecito aspettarsi:

$$T(n) = \Theta(n \cdot log(n)).$$

Stiamo risolvendo: $T(n) = 2 \cdot T(\frac{n}{2}) + n$ **(di nuovo)**

Soluzione di ricorrenze: il Master Theorem

Usiamo il teorema per la ricorrenza $T(n) = 4 \cdot T(\frac{n}{2}) + n^2 \cdot log(n)$.
Abbiamo, nuovamente, che il caso corretto è il numero 2, con $a = 4$, $b = 2$, e $k = 1$, $f(n) = n^2 \cdot log(n)$, per ottenere:

$$T(n) = \Theta(n^2 \cdot log^2(n)).$$

Soluzione di ricorrenze: se il Master Theorem non basta

Cosa accade se la ricorrenza **non** ha la forma corretta per il Master Theorem? Le ragioni possono essere tante, tra cui l'uso di arrotondamenti che per qualche ragione non si vogliono togliere, o semplicemente che ci sono chiamate ricorsive su parti dell'input di lunghezza diversa. Sebbene anche in questi casi si possa sempre utilizzare lo sviluppo (ma non il teorema che generalizza la tecnica), a volte, in letteratura, si trovano ricorrenze risolte per **sostituzione**. Questa tecnica consiste nell'**indovinare** il risultato e poi dimostrare che questo è corretto con l'induzione.

Vediamo adesso alcuni esempi di sostituzione.

Soluzione di ricorrenze: sostituzione

Possiamo come primo esempio **verificare** la soluzione di $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ attraverso una sostituzione. Questo significa fare una **ipotesi** che spiega $T(n) = O(log(n))$, e poi verificarla per induzione. Domandiamoci: perchè $T(n) = O(log(n))$? Ci sono tante possibili ipotesi:

- $T(n) \leq c \cdot log(n)$ per qualche $c > 0$;
- $T(n) \leq c \cdot log(n) \pm d$ per qualche $c, d > 0$;
- $T(n) \leq c \cdot log(n \pm d)$ per qualche $c, d > 0$;
- $\ldots$

Stiamo risolvendo: $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ **(di nuovo)**

Vale che se l'assunto è vero allora $T(n) = O(f(n))$

Immaginiamo ad esempio che $T(n) \leq c \cdot log(n)$; allora, si ha:

$$
\begin{array}{lll}
T(n) & = T(\lceil \frac{n}{2} \rceil) + 1 & \\
T(n) & \leq c \cdot log(\lceil \frac{n}{2} \rceil) + 1 & \textit{ipotesi induttiva} \\
 & \leq c \cdot log(\frac{n+1}{2}) + 1 & \textit{maggiorazione} \\
 & \leq c \cdot log(n+1) - c \cdot log(2) + 1 & \textit{proprietà logaritmi} \\
 & \leq^{?} c \cdot log(n) & \textit{NON FUNZIONA!}
\end{array}
$$

Non funziona perchè non c'è nessuna scelta di c che rende vero $c \cdot log(n+1) - c + 1 \leq c \cdot log(n)$.

è minore di $cg(n)$!!!

Stiamo risolvendo: $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$

Non esiste una strategia sicura per costruire la giusta ipotesi. Immaginiamo questa volta, per la stessa ricorrenza, che $T(n) \leq c \cdot log(n-2)$; allora, si ha:

$$
\begin{array}{lll}
T(n) & = T(\lceil \frac{n}{2} \rceil) + 1 & \\
T(n) & \leq c \cdot log(\lceil \frac{n}{2} \rceil - 2) + 1 & \textit{ipotesi induttiva} \\
 & \leq c \cdot log(\frac{n+1}{2} + \frac{1}{2} - 2) + 1 & \textit{maggiorazione} \\
 & \leq c \cdot log(\frac{n-2}{2}) + 1 & \textit{algebra} \\
 & \leq c \cdot log(n-2) - \boxed{c \cdot log(2) + 1} & \textit{algebra} \\
 & \leq c \cdot log(n-2) & \textit{maggiorazione}
\end{array}
$$

è vero cus

$c \lg_2(2) - 1 \geq 0$? $\Rightarrow c \cdot 1 \geq -1$

$\boxed{c \geq -1}$

si

Stiamo risolvendo: $T(n) = T(\lceil \frac{n}{2} \rceil) + 1$ **(di nuovo)**

Soluzione di ricorrenze: sostituzione

Come altro esempio, cerchiamo di mostrare che la soluzione di $T(n) = 2 \cdot T(\lfloor \frac{n}{2} \rfloor) + n$ che abbiamo dato (che era $T(n) = O(log(n))$) é corretta. Assumiamo che $T(n) \leq c \cdot n \cdot log(n)$ per qualche $c > 0$, e si ha:

$$
\begin{array}{lll}
T(n) & = 2 \cdot T(\lfloor \frac{n}{2} \rfloor) + n & \textit{ricorrenza} \\
 & \leq 2 \cdot (c \cdot \lfloor \frac{n}{2} \rfloor \cdot log(\lfloor \frac{n}{2} \rfloor)) + n & \textit{ipotesi induttiva} \\
 & \leq c \cdot n \cdot log(\frac{n}{2}) + n & \textit{algebra e maggiorazione} \\
 & = c \cdot n \cdot log(n) - c \cdot n \cdot log(2) + n & \textit{proprietà di } log() \\
 & = c \cdot n \cdot log(n) - c \cdot n + n & log(2) = 1 \\
 & \leq c \cdot n \cdot log(n) & \textit{maggiorazione, con } c \geq 1
\end{array}
$$

Soluzione di ricorrenze: se il Master Theorem non basta

Adesso che abbiamo capito come funziona la sostituzione, proviamo ad applicarla ad un caso reale di ricorrenza che non può essere risolta dal Master Theorem, diversamente dai casi di esempio presentati prima.

Soluzione di ricorrenze: se il Master Theorem non basta

Nel caso della ricorrenza $T(n) = T(\frac{n}{3}) + T(\frac{2}{3} \cdot n) + n$, usiamo il seguente sviluppo:

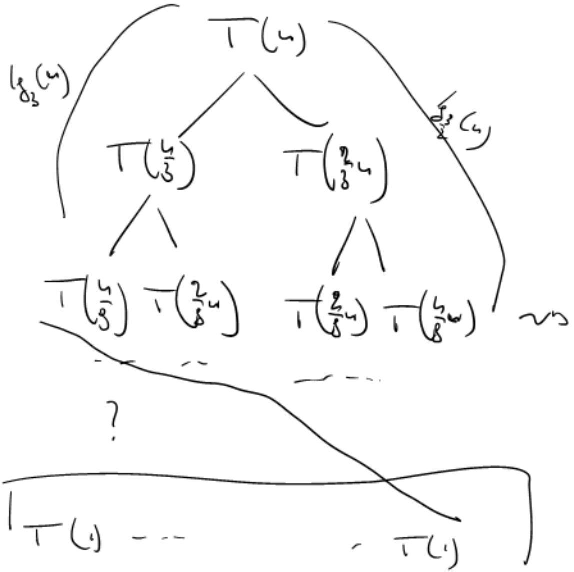
$$
\begin{aligned}
T(n) &= T(\tfrac{n}{3}) + T(\tfrac{2 \cdot n}{3}) + n \\
&= \underbrace{T(\frac{n}{9}) + T(\frac{2 \cdot n}{9}) + \frac{n}{3}}_{} + \underbrace{T(\frac{2 \cdot n}{9}) + T(\frac{4 \cdot n}{9}) + \frac{2 \cdot n}{3}}_{} + n = \\
&= \underbrace{T(\frac{n}{27}) + T(\frac{2 \cdot n}{27}) + \frac{n}{9}}_{} + \underbrace{2 \cdot (T(\frac{2 \cdot n}{27}) + T(\frac{4 \cdot n}{27}) + \frac{2 \cdot n}{9})}_{} + \underbrace{T(\frac{4 \cdot n}{27}) + T(\frac{8 \cdot n}{27}) + \frac{4 \cdot n}{27}}_{} + n + n = \\
&\dots
\end{aligned}
$$

Sotto le solite ipotesi, si ottiene

$$
\begin{aligned}
T(n) &= 2^{log_{\frac{3}{2}}(n)} + \Sigma_{i=0}^{log_{\frac{3}{2}}(n)-1} n \\
&= n^{log_{\frac{3}{2}}(2)} + n \cdot \Sigma_{i=0}^{log_{\frac{3}{2}}(n)-1} 1 = \underbrace{n^{log_{\frac{3}{2}}(2)}}_{\text{Casi base}} + \underbrace{n \cdot log_{\frac{3}{2}}(n)}_{\text{Casi induttivi}}.
\end{aligned}
$$

Stiamo risolvendo: $T(n) = T(\frac{n}{3}) + T(\frac{2}{3} \cdot n) + n$

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



$\leadsto$ n

$\leadsto$ $\frac{n}{3} + \frac{2}{3}n = n$

$\leadsto$ n

$$T(n) \leq \sum_{i=0}^{\lg_{\frac{3}{2}}(n)-1} n + 2^{\lg_{\frac{3}{2}}(n)}$$

Soluzione di ricorrenze: se il Master Theorem non basta

In questo esempio stiamo facendo l'ipotesi che tutti i rami di ricorsione siano lunghi quanto il piú lungo di essi (alcuni rami dividono per 3, altri dividono per 2 terzi, quindi in realtá non è cosí). Inoltre siamo giunti ad un'espressione non facile da valutare. Una strategia, a questo punto, potrebbe essere ipotizzare che valga:

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

e utilizzare la sostituzione per verificarlo. In questo caso **non** siamo autorizzati ad utilizzare la notazione $\Theta()$ a meno di non utilizzare la sostituzione una seconda volta.

Soluzione di ricorrenze: se il Master Theorem non basta

Per assicurarci dunque che la soluzione di $T(n) = T(\frac{n}{3}) + T(\frac{2}{3} \cdot n) + n$ sia corretta, assumiamo che $T(n) \leq c \cdot n \cdot log(n)$ per qualche $c > 0$, e si ha:

1

$$
\begin{array}{lll}
T(n) & = T(\frac{n}{3}) + T(\frac{2}{3} \cdot n) + n & \textit{ricorrenza} \\
 & \leq c \cdot \frac{n}{3} \cdot log(\frac{n}{3}) + c \cdot \frac{2 \cdot n}{3} \cdot log(\frac{2 \cdot n}{3}) + n & \textit{ipotesi induttiva} \\
 & \leq c \cdot \frac{n}{3} \cdot (log(n) - log(3)) + c \cdot \frac{2 \cdot n}{3} \cdot (log(2) + log(n) - log(3)) + n & \textit{proprietà di } log() \\
 & \leq c \cdot \frac{n}{3} \cdot log(n) - c \cdot \frac{n}{3} \cdot log(3) + c \cdot \frac{2 \cdot n}{3} + c \cdot \frac{2 \cdot n}{3} \cdot log(n) - c \cdot \frac{2 \cdot n}{3} \cdot log(3) + n & \textit{algebra} \\
 & \leq c \cdot n \cdot log(n) - \underbrace{(c \cdot n \cdot log(3) - c \cdot \frac{2 \cdot n}{3} - n)}_{\geq 0} &
\end{array}
$$

$\leq c\, n\, lg(n)$

Questo ci porta a esprimere una condizione su c (ricordiamo che $log(3) = 1, \ldots$, mentre $\frac{2}{3} = 0, 6 \ldots$); poichè questa condizione è compatibile con $c > 0$, la proprietà è verificata e possiamo concludere che $T(n) = O(n \cdot log(n))$.

Stiamo risolvendo: $T(n) = T(\frac{n}{3}) + T(\frac{2}{3} \cdot n) + n$

$$c\cancel{x}\lg(3) - c\frac{2}{3}\cancel{x} - \cancel{x} \geq 0$$

$$c\lg(3) - c\frac{2}{3} - 1 \geq 0$$

$$c\left(\lg(3) - \frac{2}{3}\right) \geq 1$$

$$\boxed{c \geq \frac{1}{\lg(3) - \frac{2}{3}}}$$

CONDIZIONE NECESTABILE SU c (POSITIVO)

Soluzione di ricorrenze: riassunto

Una strategia generale per una ricorrenza puó essere delinata come segue: come primo step proviamo il Master Theorem, e se non funziona (o la forma della ricorrenza non lo permette), sviluppiamo in serie per poi verificare, se necessario (oppure, se lo sviluppo è troppo complesso e sono stati necessari dei passi di approssimazione), il risultato con una o piú sostituzioni.

$\boxed{\text{ESERCIZIO 20}}$: $T(n) = 4T\left(\frac{n}{2}\right) + \underbrace{3n^2 - 17}_{\Theta(n^2)}$

$T(n) = \underset{a}{4}\, T\left(\frac{n}{\underset{b}{2}}\right) + \Theta(n^2)$

$n^{\log_b(a)} = n^{\log_2(4)} = n^2$

$f(n) = \Theta(n^2) \overset{?}{=}$

$O\left(n^{\log_b(a) - \varepsilon}\right)$

$\Theta\left(n^{\log_b(a)}\right) \Rightarrow$ caso II $(k = 0)$

$\Omega\left(n^{\log_b(a) + \varepsilon}\right)$ $\Rightarrow T(n) = \Theta\left(n^2 \log(n)\right)$

$\boxed{A}$

$$\boxed{C}$$

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

$T(n)$ $\rightsquigarrow$ n^3

$T\left(\frac{n}{2}\right)$ $T\left(\frac{n}{2}\right)$ $\rightsquigarrow$ $\frac{n^3}{8} + \frac{n^3}{8} = \frac{2}{8} n^3 = \frac{1}{4} n^3$

$T\left(\frac{n}{4}\right)$ $T\left(\frac{n}{4}\right)$ -- $\rightsquigarrow$ $4 \cdot \frac{n^3}{64} = \frac{1}{16} n^3$

$\underbrace{T(1) \;\text{--}\; T(1)}$

$$\sum_{i=0}^{\lg_2(n)} \left(\frac{1}{4}\right)^i n^3 + 2^{\lg_2(n)} =$$

$$n^3 \cdot c + n = \Theta(n^3)$$

$\boxed{\text{ESERCIZIO 29}}$

$$T(n) = 7T\left(\frac{n}{2}\right) + n^2$$

$$T'(n) = aT'\left(\frac{n}{4}\right) + n^2$$

$a = 7$

$b = 2$ $\Rightarrow$ $\lg_b(a) = \lg_2(7)$

$f(n) = n^2$

$$f(n) = n^2 = O\left(n^{\lg_2(7) - \varepsilon}\right)$$

$\Rightarrow$ CASO I $\Rightarrow T(n) = \Theta\left(n^{\lg_2(7)}\right)$

$a = a$

$b = 4$

$f(n) = n^2$

$\lg_4(a)$

$$f(n) = O\left(n^{\lg_4(a) - \varepsilon}\right)$$

$\Downarrow$

$$T'(n) = \Theta\left(n^{\lg_4(a)}\right)$$

RICONDA!: IL CONFRONTO CON IL CASO I DI $T'(n)$

Condiții

Rezolv:

$$\lg_4(a) < \lg_2(7) \iff \lg_4(a) < \frac{\lg_4(7)}{\lg_4(2)}$$

$$\iff \lg_4(a) < 2\lg_4(7)$$

$$\iff a < 7^2 \Rightarrow a < 49$$

$$\Rightarrow a = 48$$

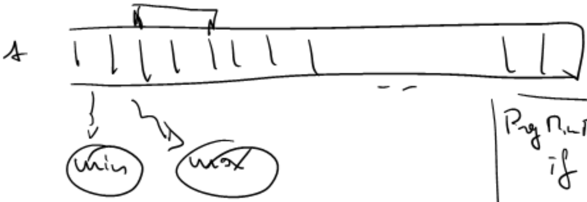
$\boxed{C}$

Conclusione

Le ricorrenze sono un problema molto generale. É improbabile che sviluppando algoritmi nuovi emergano ricorrenze molto complesse, sebbene questo succeda di tanto in tanto. La soluzione di ricorrenze nel caso generale è, invece, interessante da un punto di vista matematico.

Esercizio 17

1D



Totale confronti: $\frac{3}{2} n$

MinMax(A)
  if A[1] < A[2] then
    min = A[1]
    max = A[2]
  else
    min = A[2]
    max = A[1]
for (i = 3 to A.length step 2)
  if (A[i] < A[i+1])
    if A[i] < min then min = A[i]
    if (A[i+1]) > max then max = A[i+1]
  else
  ...

ESERCIZIO 18

$$A = \begin{bmatrix} 5 & 10 & 12 & 24 & 30 \\ 7 & 11 & 13 & 25 & 31 \\ 10 & 14 & \cdots & & \\ \cdots & & & & \end{bmatrix}$$

$n \times n$

Problema: trova $\underline{k \text{ in } A}$

Soluzione (1): guardo tutti gli elementi

complessità: $\Theta(n^2)$

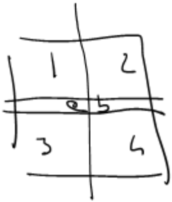
Conclusioni

Sol (2): Applica Binary Search a tutte le righe

Complessità: $\Theta(n \lg(n))$

Sol (3): Modifica Bin Search

A



$\rightarrow$ riga centrale

$k < a \Rightarrow$ NO 4

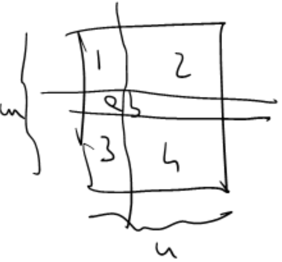
$k > b \Rightarrow$ NO 1

$a \leq k \leq b \Rightarrow$ NO 1, 4

Complessità: $T(n) = 3T\left(\frac{n}{2}\right) + 1 = \Theta\left(n^{1.58}\right)$

caso pessimo

SOLUZ (4):



k [illegible] a, b
([illegible])

$\Rightarrow$ NO $1, 4$

$$T(n,m) = T\left(\frac{n}{2}, 1\right) + T\left(\frac{n}{2}, (m-1)\right) + O(\lg(n))$$

3

=