

Algoritmi e Strutture Dati

Valutazione del tempo di esecuzione degli algoritmi

Stima del limite asintotico superiore

- Nelle prossime lezioni definiremo un semplice metodo per **stimare il limite asintotico superiore** $O(\cdot)$ del tempo di esecuzione di **algoritmi iterativi**.
 - Stabilire il limite superiore per le operazioni elementari
 - Stabilire il limite superiore per le strutture di controllo
- Ci da un limite superiore che funge da stima, **non garantisce** di trovare la **funzione precisa del tempo di esecuzione**.

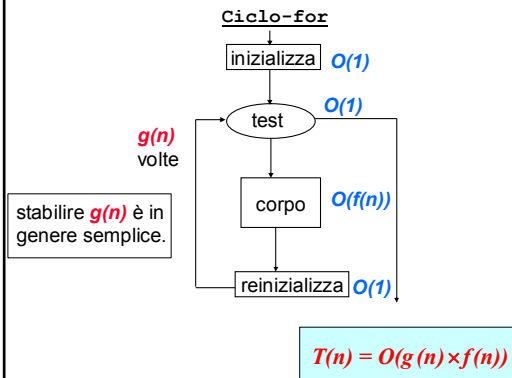
Tempo di esecuzione: operazioni semplici

Operazioni Semplici

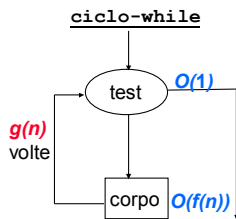
- operazioni aritmetiche (+, *, ...)
- operazioni logiche (&&, ||, ...)
- confronti (\leq , \geq , =, ...)
- assegnamenti (a = b) senza chiamate di funzione
- operazioni di lettura (read)
- operazioni di controllo (break, continue, return)

$$T(n) = \Theta(1) \Rightarrow T(n) = O(1)$$

Tempo di esecuzione: ciclo for



Tempo di esecuzione: ciclo while



Bisogna stabilire un limite per il numero di iterazioni del ciclo, $g(n)$.
Può essere necessaria una prova induttiva per $g(n)$.

$$T(n) = O(g(n) \times f(n))$$

Ciclo while: esempio

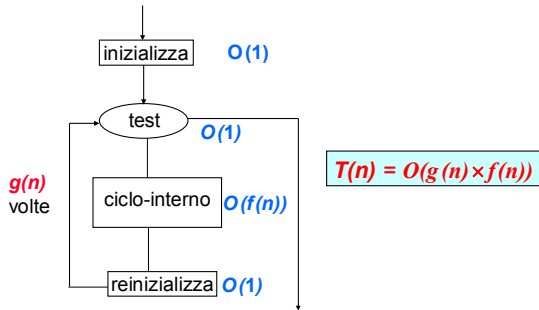
Ricerca dell'elemento x all'interno di un array $A[1..n]$:

```
i = 0 (1)
while x ≠ A[i] (2)
    i = i+1 (3)
```

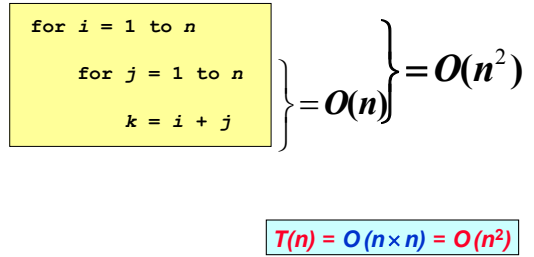
(1) $O(1)$
test in (2) $O(1)$
(3) $O(1)$
iterazioni massimo n

$$O(\text{ciclo-while}) = O(1) + n O(1) = O(n)$$

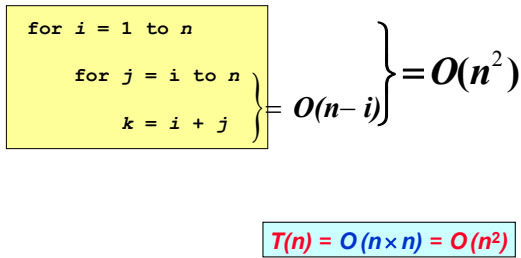
Tempo di esecuzione: cicli innestati



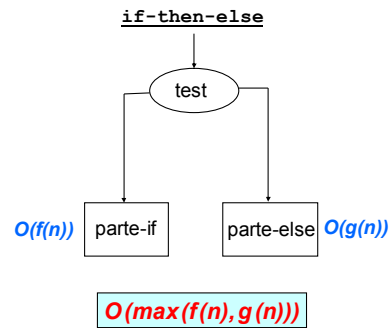
Cicli innestati: esempio



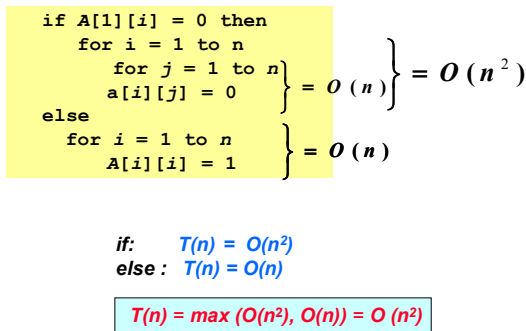
Cicli innestati: esempio



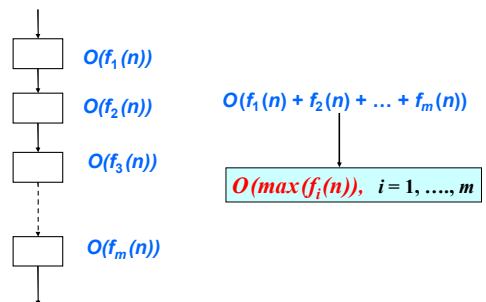
Tempo di esecuzione: If-Then-Else



If-Then-Else: esempio



Tempo di esecuzione: blocchi sequenziali



Blocchi sequenziali: esempio

```
for i = 1 to n } = O(n)
  A[1] = 0
for i = 1 to n } = O(n)
  for j = 1 to n } = O(n)
    A[i] = A[i] + A[i]
```

$$T(n) = O(\max(f(\text{ciclo-1}), f(\text{ciclo-2}))) \\ = O(\max(n, n^2)) \\ = O(n^2)$$

Esempio: Insert Sort

```
InsertSort(array A[1..n])
  for j = 2 to n
    key = A[j] = O(1)
    i = j - 1 = O(1)
    while i > 0 and A[i] > key } = O(n)
      A[i+1] = A[i]
      i = i - 1
    A[i+1] = key = O(1)
```

$$T(n) = O(g(n) \times \max(1, 1, n, 1)) \\ = O(n \times n) \\ = O(n^2)$$

Tempo di esecuzione di algoritmi ricorsivi

- E per gli algoritmi ricorsivi?
- Il tempo di esecuzione è espresso tramite una equazione di ricorrenza.

Esempio:

$$\text{Merge Sort: } T(n) = \begin{cases} \Theta(1) & \text{se } n=1 \\ 2T(n/2) + \Theta(n) & \text{se } n>1 \end{cases}$$

- Sono necessarie tecniche specifiche per risolvere le equazioni di ricorrenza

Algoritmi e Strutture Dati

Tempo di esecuzione di algoritmi ricorsivi

Tempo di esecuzione per algoritmi ricorsivi

Esempio: Fattoriale

```
fact(int n)
  if n <= 1 then
    return 1 /* Caso Base
  else
    return n*fact(n-1) /* Passo Induttivo
```

$$T(n) = \begin{cases} O(1) & \text{se } n=1 \\ O(1) + T(n-1) & \text{se } n>1 \end{cases}$$

Soluzione di equazioni di ricorrenza

- Esistono molto metodi. Ne mostreremo tre:

Il Metodo Iterativo

- Si itera la regola induttiva di $T(n)$ in termini di n e del caso base.
- Richiede manipolazione delle somme

Il Metodo di Sostituzione

- Si ipotizza una possibile soluzione
- Si sostituisce l'ipotetica soluzione nei casi base e induttivo
- Si dimostra la correttezza della ipotesi tramite induzione matematica

Il Metodo Principale

Il Metodo Iterativo

Base: $T(1) = a$
 Induzione: $T(m) = b + T(m-1)$

I. Sostituire ad m i valori $n, n-1, n-2 \dots$ finché si ottiene il caso base

- 1) $T(n) = b + T(n-1)$ sostituire m con n
- 2) $T(n-1) = b + T(n-2)$ sostituire m con $n-1$
- 3) $T(n-2) = b + T(n-3)$ sostituire m con $n-2$
-
- $n-1$). $T(2) = b + T(1)$ sostituire m con 2
- $T(1) = a$ noto

Il Metodo Iterativo

II. Sostituire $T(n-1), T(n-2) \dots$ fino al caso base e sostituirlo.

$$\begin{aligned} T(n) &= b + T(n-1) &= \\ &= b + b + T(n-2) &= 2*b + T(n-2) = \\ &= b + b + b + T(n-3) &= 3*b + T(n-3) = \\ &= b + b + b + b + T(n-4) &= 4*b + T(n-4) = \end{aligned}$$

$$T(n) = \sum_{i=1}^{n-1} b + T(1) = (n-1) \cdot b + T(1)$$

Inserire il caso base

$$T(n) = (n-1) \cdot b + a$$

III. Valutare l'espressione O-grande associata

$$T(n) = b*n - b + a = O(n)$$

Il Metodo iterativo: Fattoriale

Esempio: Fattoriale

```
fact(int n)
  if n <= 1 then
    return 1          /* Caso Base
  else
    return n*fact(n-1) /* Passo Induttivo
```

$$T(n) = \begin{cases} O(1) & \text{se } n = 1 \\ O(1) + T(n-1) & \text{se } n > 1 \end{cases}$$

Equazione di ricorrenza

Base: $T(1) = a$
 Induzione: $T(m) = b + T(m-1)$

Il Metodo iterativo: Fattoriale

Analisi di fact

Caso Base: $T(0) = O(1)$,
 $T(1) = O(1)$

Passo Induttivo: $O(1) + \max(O(1), T(n-1))$
 $O(1) + T(n-1)$, per $n > 1$

Per il fattoriale, l'analisi risulta $T(n) = O(n)$

Il Metodo iterativo: esempio

$$T(n) = 3 T(n/4) + n$$

Il Metodo iterativo: esempio

$$\begin{aligned} T(n) &= 3 T(n/4) + n = \\ &= 3(3 T(n/16) + n/4) + n \end{aligned}$$

Il Metodo iterativo: esempio

$$\begin{aligned} T(n) &= 3 T(n/4) + n = \\ &= 3(3 T(n/16) + n/4) + n = \\ &= 9 T(n/16) + 3 n/4 + n \end{aligned}$$

Il Metodo iterativo: esempio

$$\begin{aligned} T(n) &= 3 T(n/4) + n = \\ &= 3(3 T(n/16) + n/4) + n = \\ &= 9 T(n/16) + 3 n/4 + n = \\ &= 27 T(n/64) + 9 n/16 + 3 n/4 + n \end{aligned}$$

Il Metodo iterativo: esempio

$$\begin{aligned} T(n) &= 3 T(n/4) + n = \\ &= 3(3 T(n/16) + n/4) + n = \\ &= 9 T(n/16) + 3 n/4 + n = \\ &= 27 T(n/64) + 9 n/16 + 3 n/4 + n = \\ &..... \end{aligned}$$

Quando ci si ferma?

Il Metodo iterativo: esempio

$$\begin{aligned} T(n) &= 3 T(n/4) + n = \\ &= 3(3 T(n/16) + n/4) + n = \\ &= 9 T(n/16) + 3 n/4 + n = \\ &= 27 T(n/64) + 9 n/16 + 3 n/4 + n = \\ &..... \end{aligned}$$

Quando ci si ferma?
quando $n/(4^i) = 1$
cioè quando $i > \log_4 n$

$$T(n) < n + 3 n/4 + 9 n/16 + 27 T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

Il Metodo iterativo: esempio

$$T(n) < n + 3 n/4 + 9 n/16 + 27 T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

Contiene una serie geometrica, che è del tipo

$$\sum_{i=0}^n x^i = 1 + x + x^2 + \dots + x^n$$

$$T(n) < n + 3 n/4 + 9 n/16 + 27 T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

$$\leq n \sum_{i=0}^{\log_4 n} (3/4)^i + \Theta(n^{\log_4 3})$$

$$3^{\log_4 n} = n^{\log_4 3}$$

Il Metodo iterativo: esempio

$$T(n) < n + 3 n/4 + 9 n/16 + 27 T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

Contiene una serie geometrica, che è del tipo

$$\sum_{i=0}^n x^i = 1 + x + x^2 + \dots + x^n$$

quando $|x| < 1$ converge a $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$

$$\sum_{i=0}^{\infty} (3/4)^i = \frac{1}{1-3/4} = 4$$

$$T(n) < n + 3 n/4 + 9 n/16 + 27 T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

$$\leq n \sum_{i=0}^{\log_4 n} (3/4)^i + \Theta(n^{\log_4 3})$$

$$3^{\log_4 n} = n^{\log_4 3}$$

Il Metodo iterativo: esempio

$$T(n) < n + 3n/4 + 9n/16 + 27T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

Contiene una serie geometrica, che è del tipo

$$\sum_{i=0}^n x^i = 1 + x + x^2 + \dots + x^n$$

quando $|x| < 1$ converge a $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$

$$\sum_{i=0}^{\infty} (3/4)^i = \frac{1}{1-3/4} = 4$$

$$T(n) < n + 3n/4 + 9n/16 + 27T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

$$\leq n \sum_{i=0}^{\infty} (3/4)^i + \Theta(n^{\log_4 3}) = 4n + o(n)$$

$$3^{\log_4 n} = n^{\log_4 3} \text{ e } \log_4 3 < 1$$

Il Metodo iterativo: esempio

$$T(n) < n + 3n/4 + 9n/16 + 27T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

Contiene una serie geometrica, che è del tipo

$$\sum_{i=0}^n x^i = 1 + x + x^2 + \dots + x^n$$

quando $|x| < 1$ converge a $\sum_{i=0}^{\infty} x^i = \frac{1}{1-x}$

$$\sum_{i=0}^{\infty} (3/4)^i = \frac{1}{1-3/4} = 4$$

$$T(n) < n + 3n/4 + 9n/16 + 27T(n/64) + \dots + 3^{\log_4 n} \Theta(1)$$

$$\leq n \sum_{i=0}^{\infty} (3/4)^i + \Theta(n^{\log_4 3}) = 4n + o(n)$$

$$= O(n)$$

$$3^{\log_4 n} = n^{\log_4 3} \text{ e } \log_4 3 < 1$$

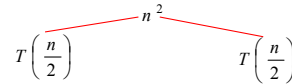
Alberi di Ricorrenza

Gli **alberi di ricorrenza** rappresentano un modo conveniente per visualizzare i passi di sostituzione necessari per risolvere una ricorrenza col **Metodo Iterativo**.

- Utili per semplificare i calcoli ed evidenziare le **condizioni limite** della ricorrenza.

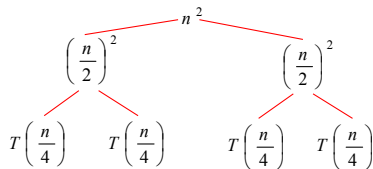
Alberi di Ricorrenza

Esempio: $T(n) = 2T(n/2) + n^2$



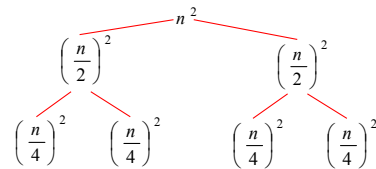
Alberi di Ricorrenza

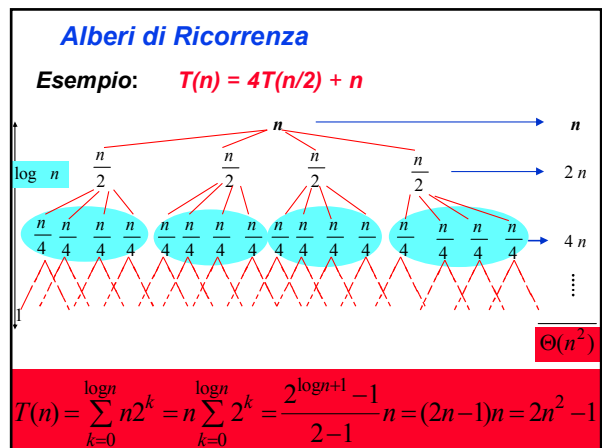
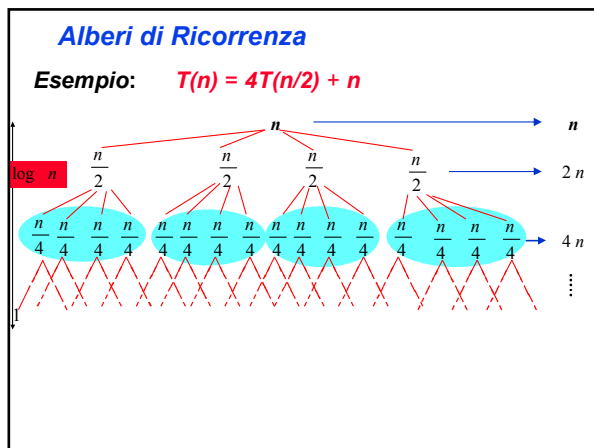
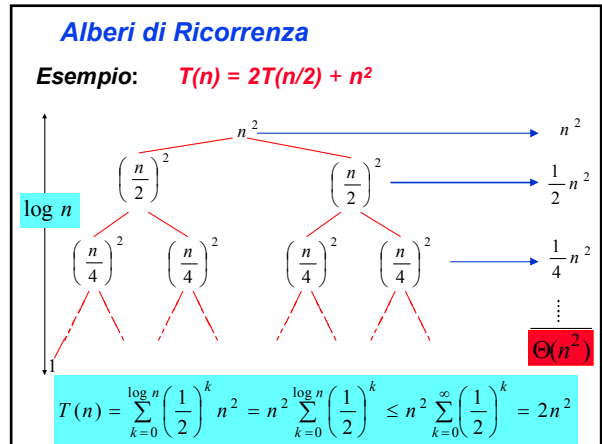
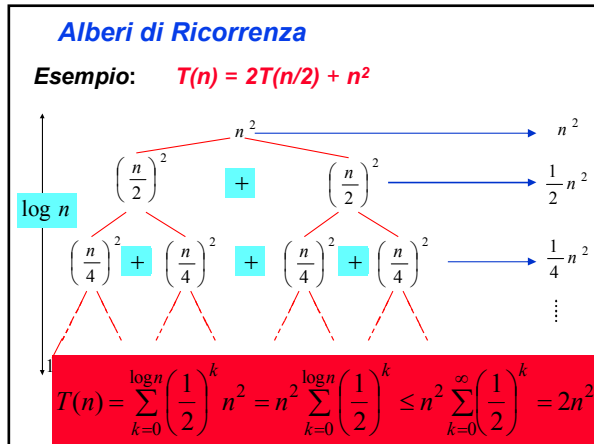
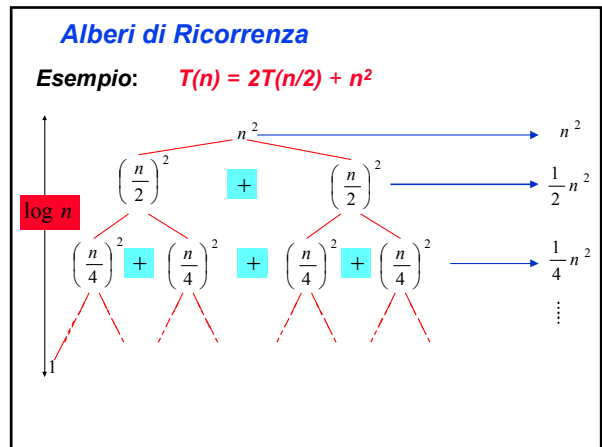
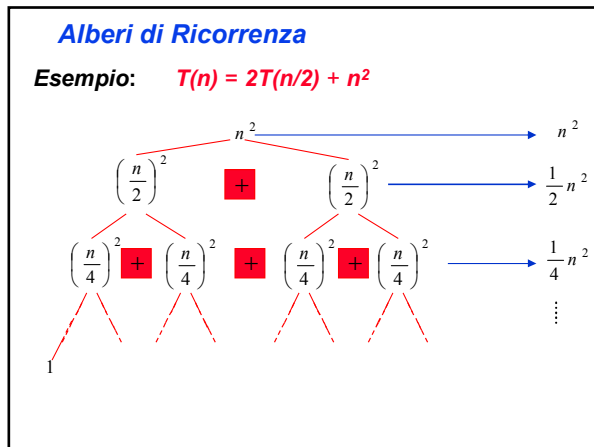
Esempio: $T(n) = 2T(n/2) + n^2$

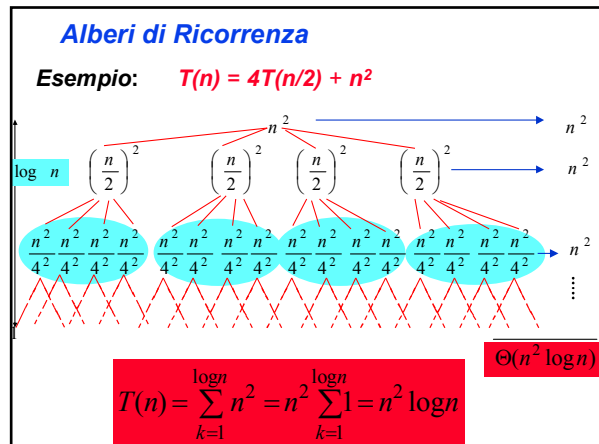
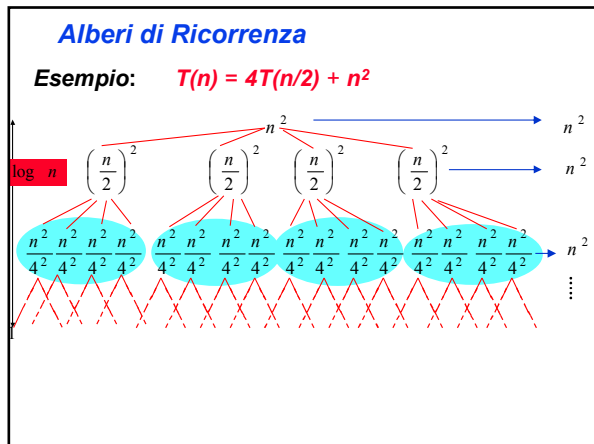


Alberi di Ricorrenza

Esempio: $T(n) = 2T(n/2) + n^2$







Metodo iterativo: conclusioni

Importante focalizzarsi su due parametri

- il numero di volte in cui la ricorrenza deve essere iterata prima di giungere alla condizione limite (o base)
- la somma dei termini che compaiono ad ogni livello della iterazione.