

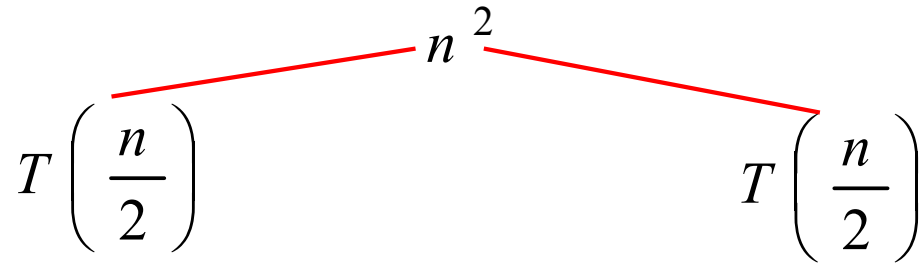
Alberi di Ricorrenza

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.

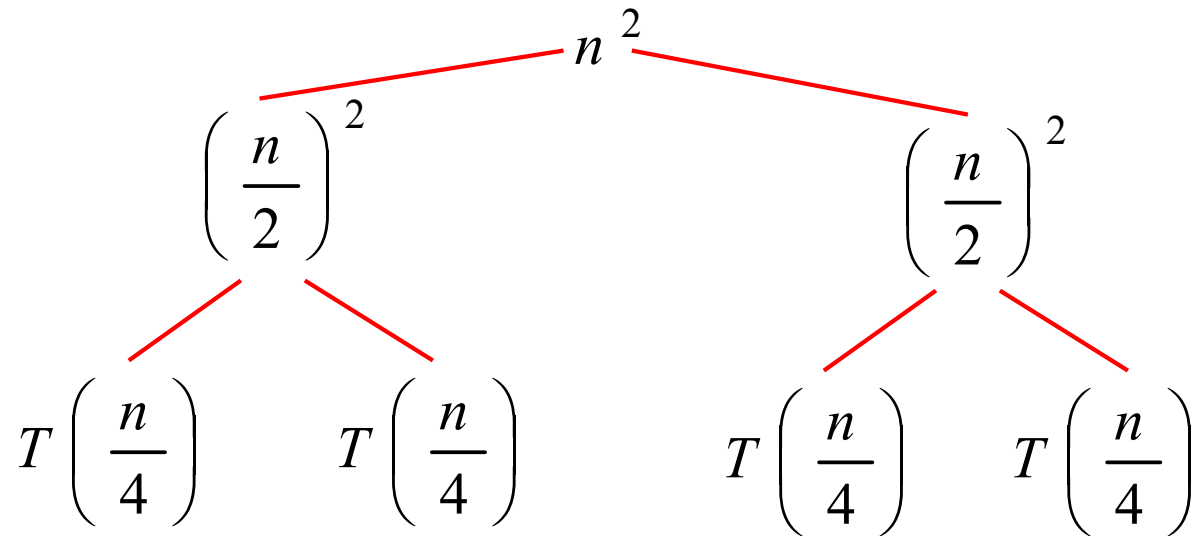
Alberi di Ricorrenza

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



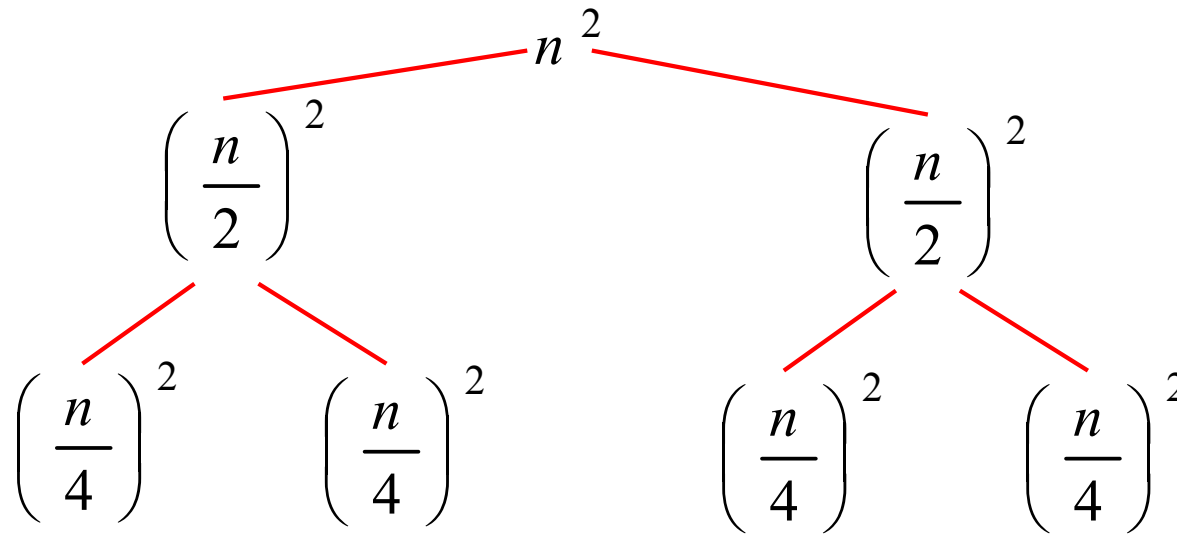
Alberi di Ricorrenza

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



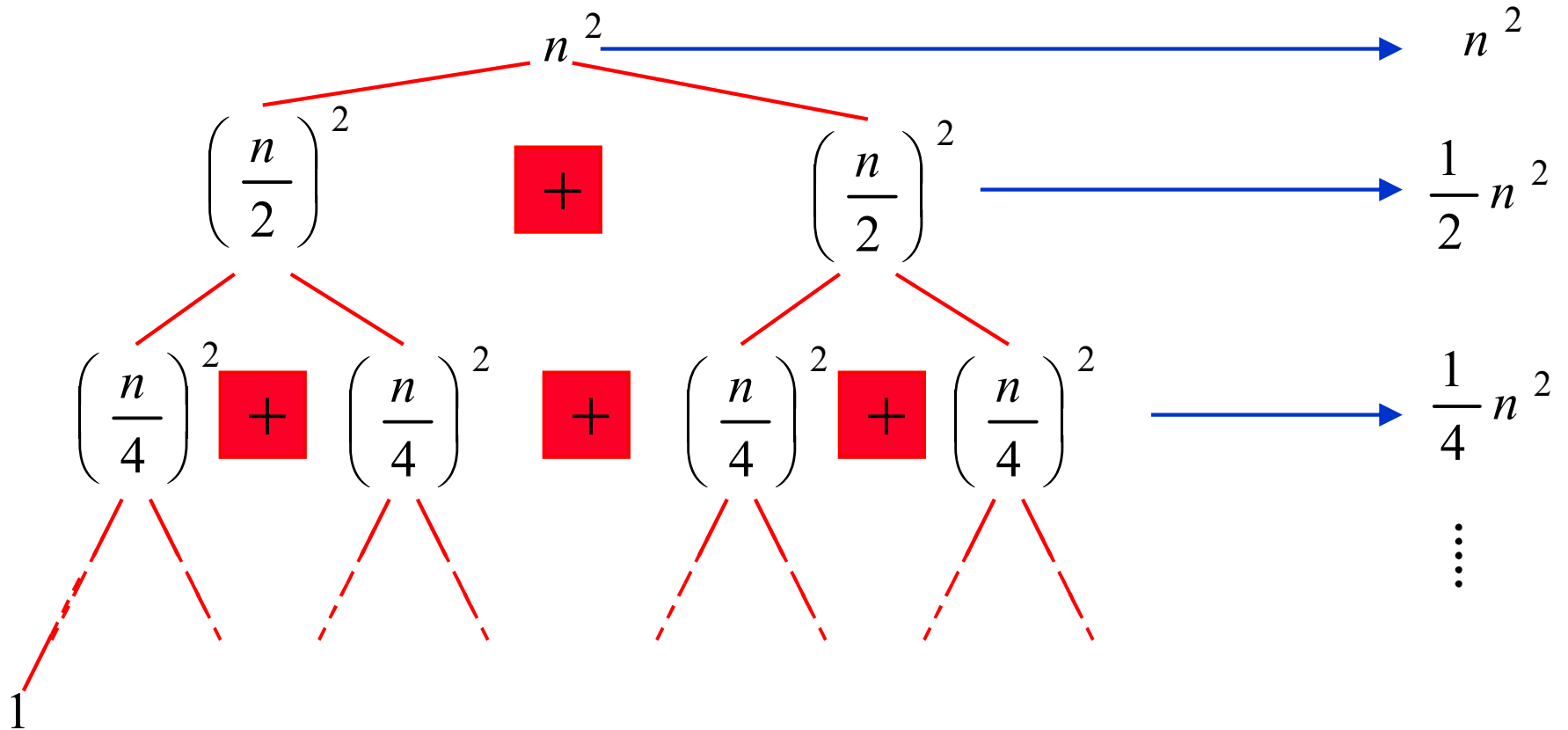
Alberi di Ricorrenza

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



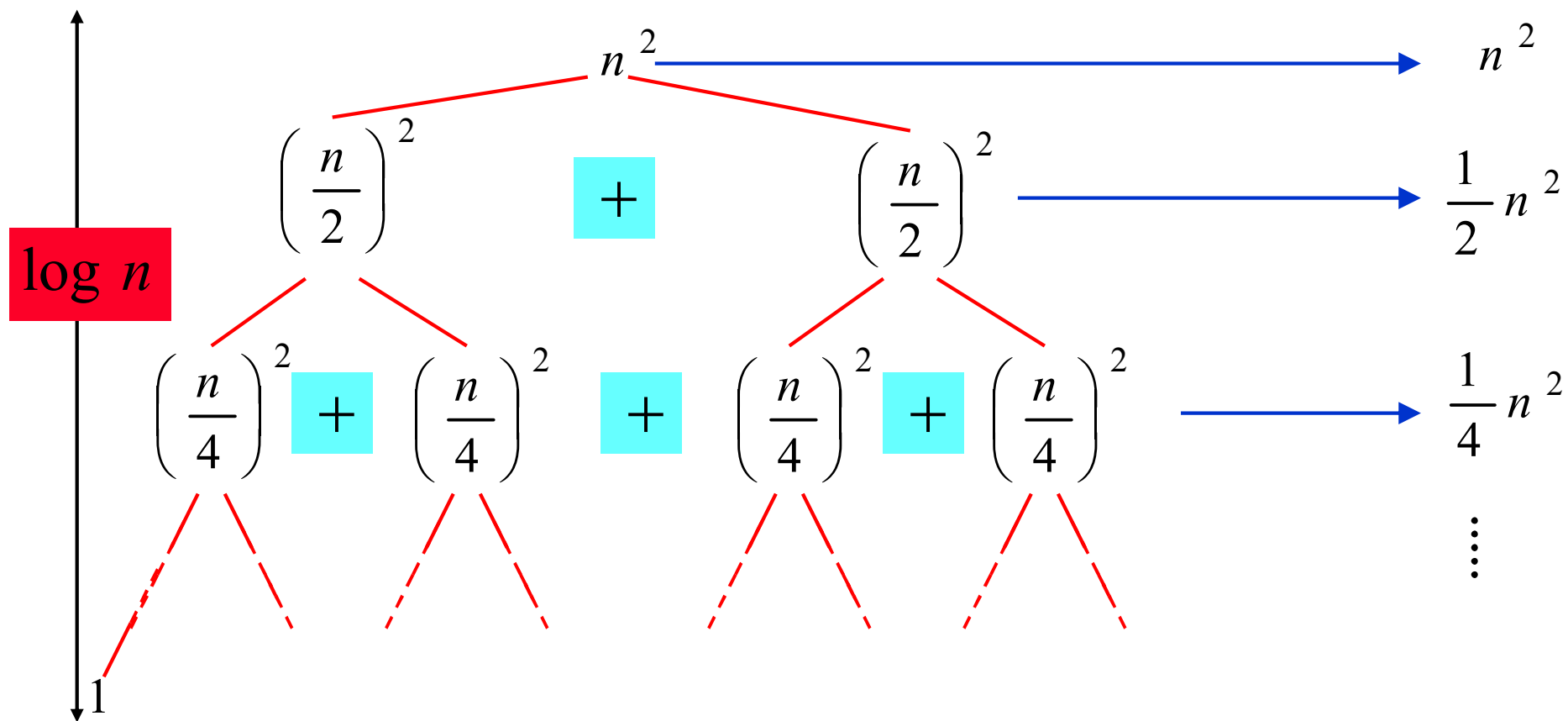
Alberi di Ricorrenza

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



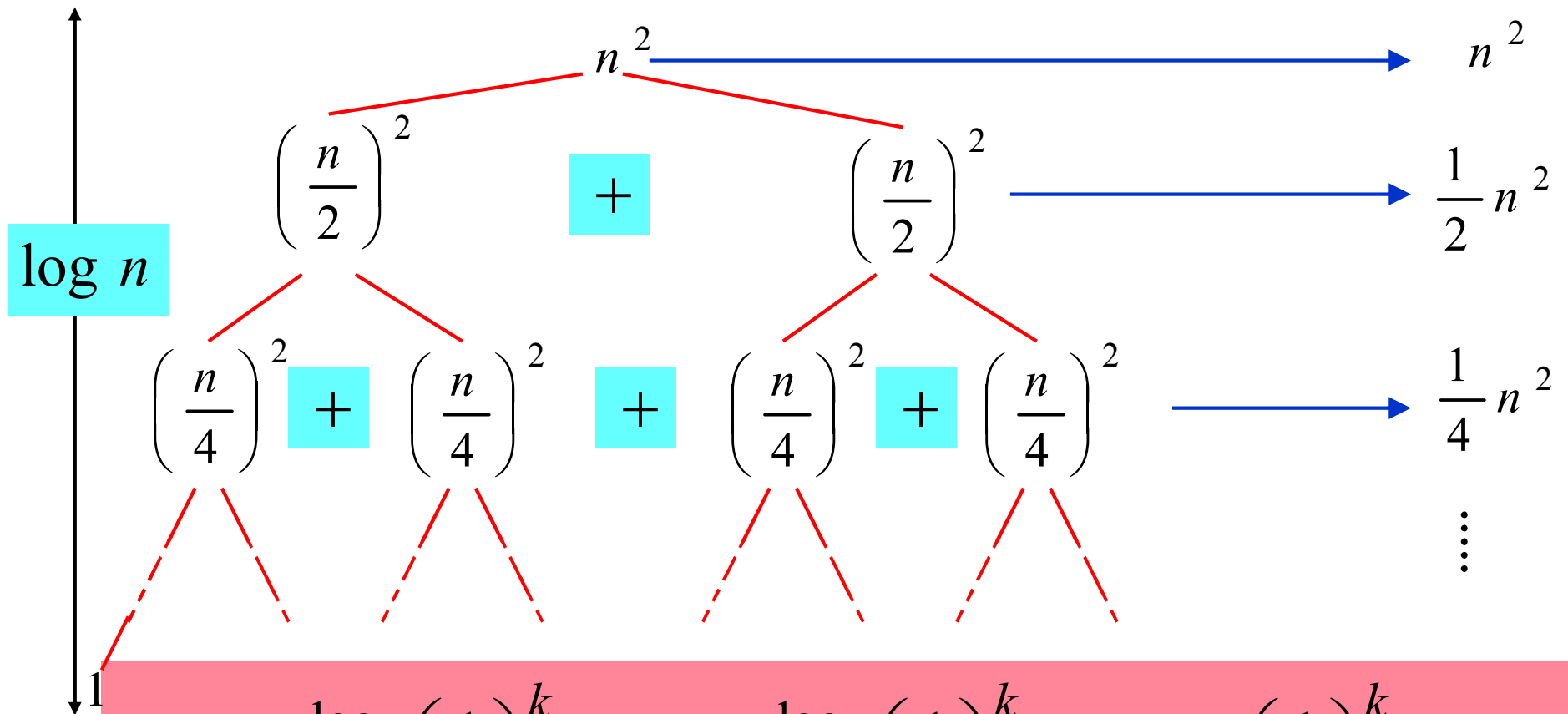
Alberi di Ricorrenza

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



Alberi di Ricorrenza

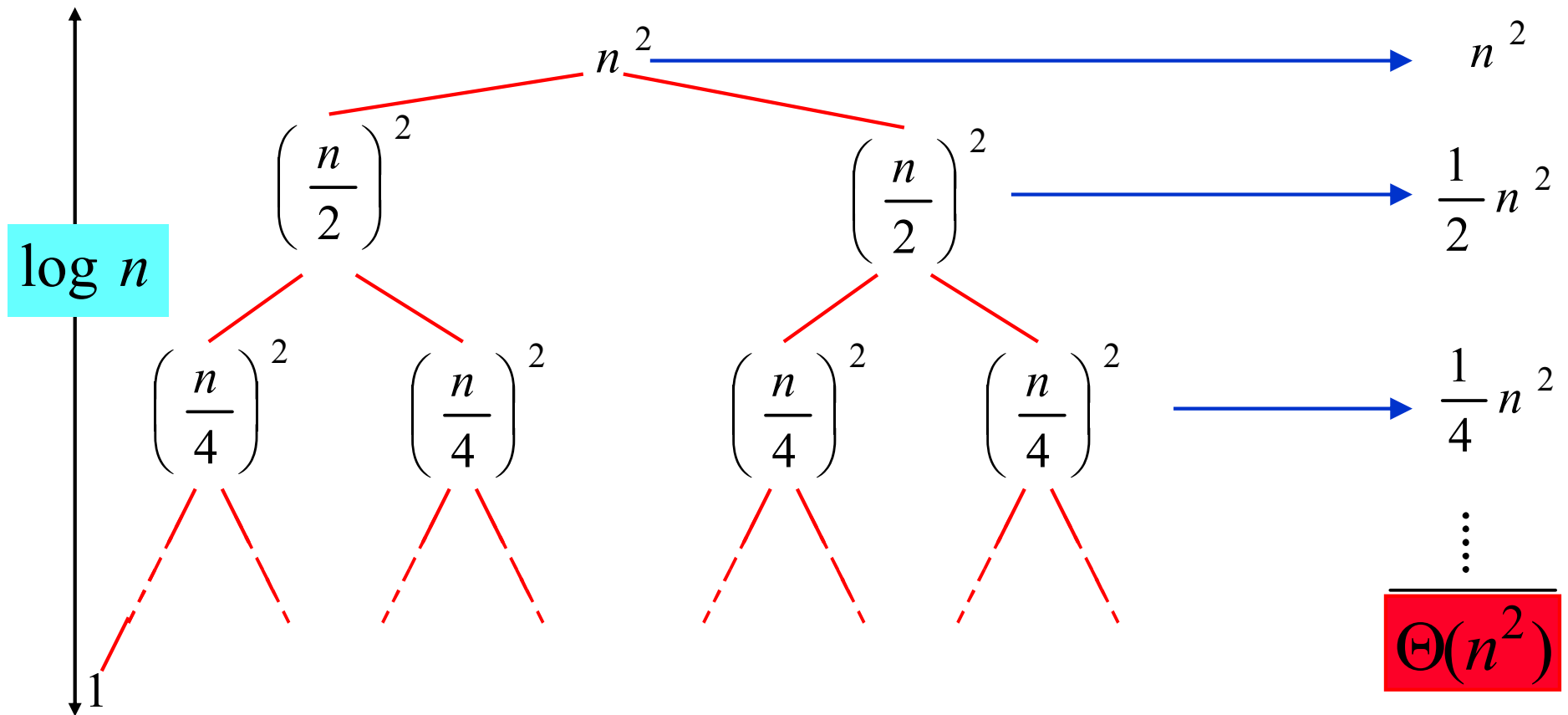
Esempio: $T(n) = 2T(n/2) + n^2$



$$T(n) = \sum_{k=0}^{\log n} \left(\frac{1}{2}\right)^k n^2 = n^2 \sum_{k=0}^{\log n} \left(\frac{1}{2}\right)^k \leq n^2 \sum_{k=0}^{\infty} \left(\frac{1}{2}\right)^k = 2n^2$$

Alberi di Ricorrenza

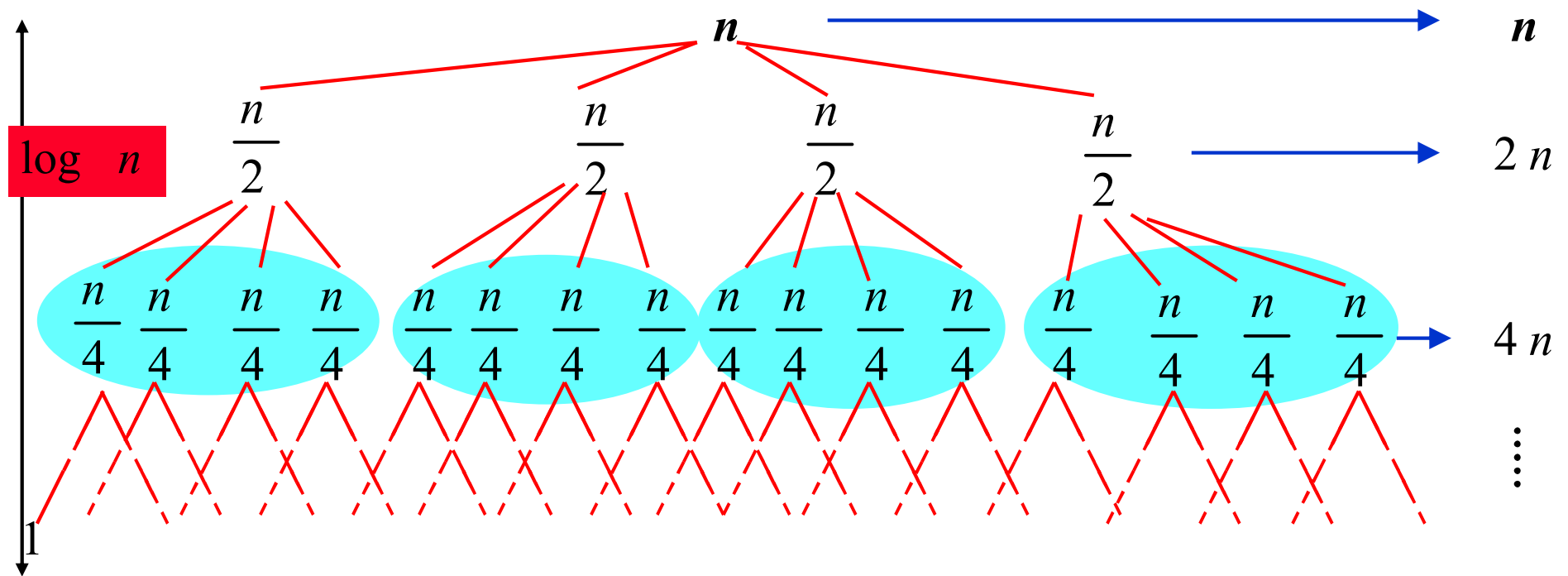
Esempio: $T(n) = 2T(n/2) + n^2$



$$T(n) = \sum_{k=0}^{\log n} \left(\frac{1}{2}\right)^k n^2 = n^2 \sum_{k=0}^{\log n} \left(\frac{1}{2}\right)^k \leq n^2 \sum_{k=0}^{\infty} \left(\frac{1}{2}\right)^k = 2n^2$$

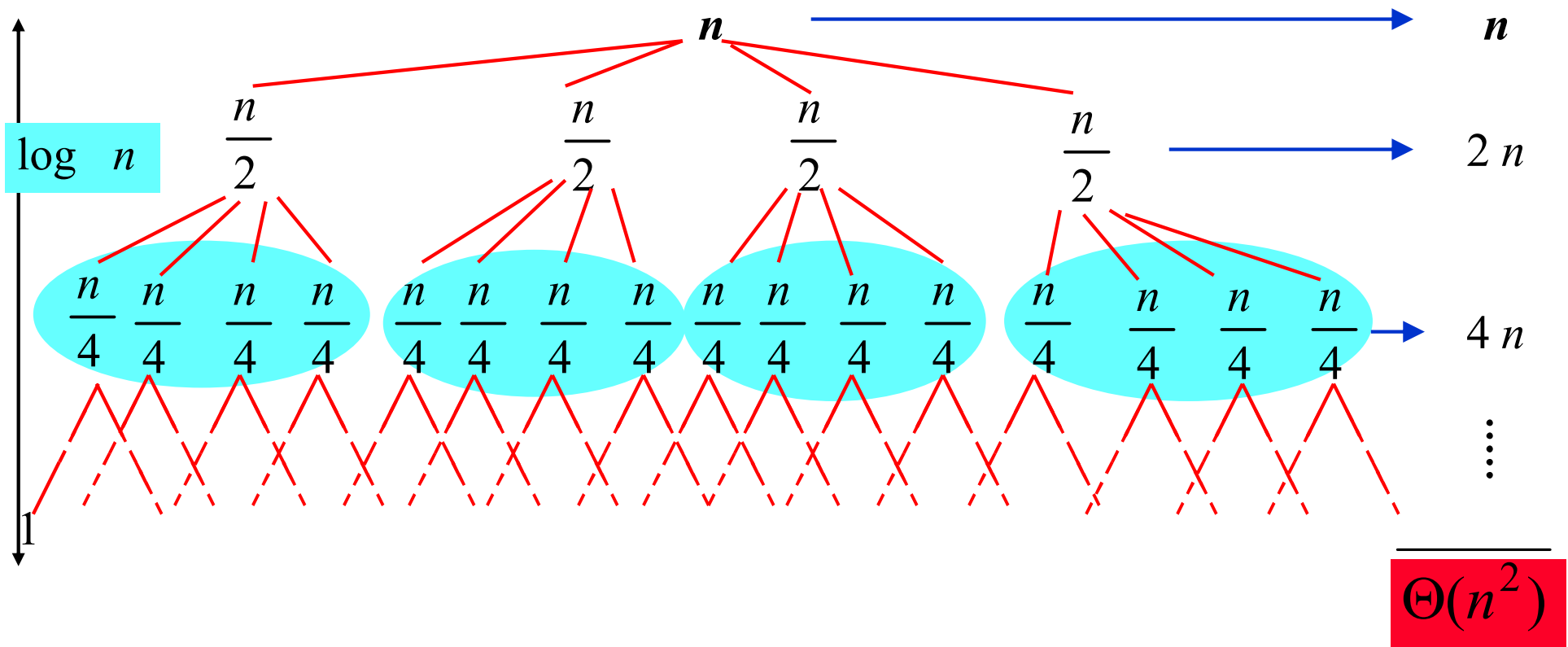
Alberi di Ricorrenza

Esempio: $T(n) = 4T(n/2) + n$



Alberi di Ricorrenza

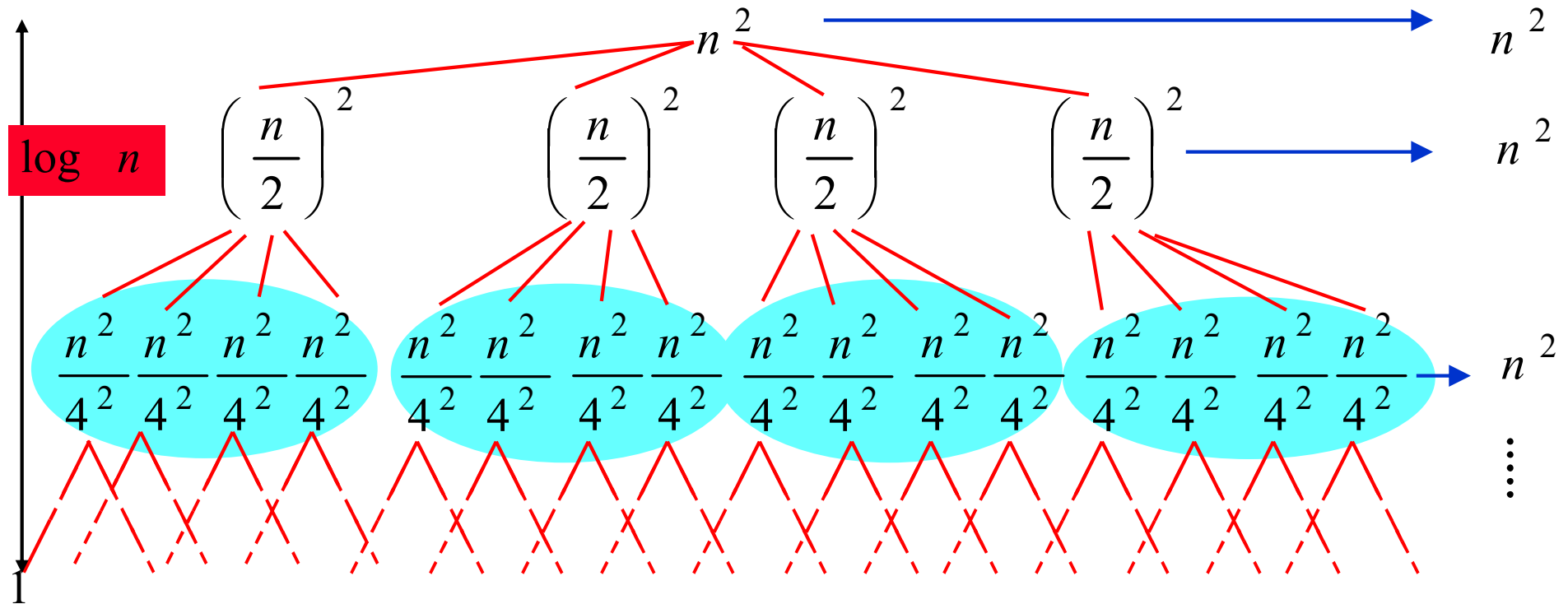
Esempio: $T(n) = 4T(n/2) + n$



$$T(n) = \sum_{k=0}^{\log n} n2^k = n \sum_{k=0}^{\log n} 2^k = \frac{2^{\log n + 1} - 1}{2 - 1} n = (2n - 1)n = 2n^2 - n$$

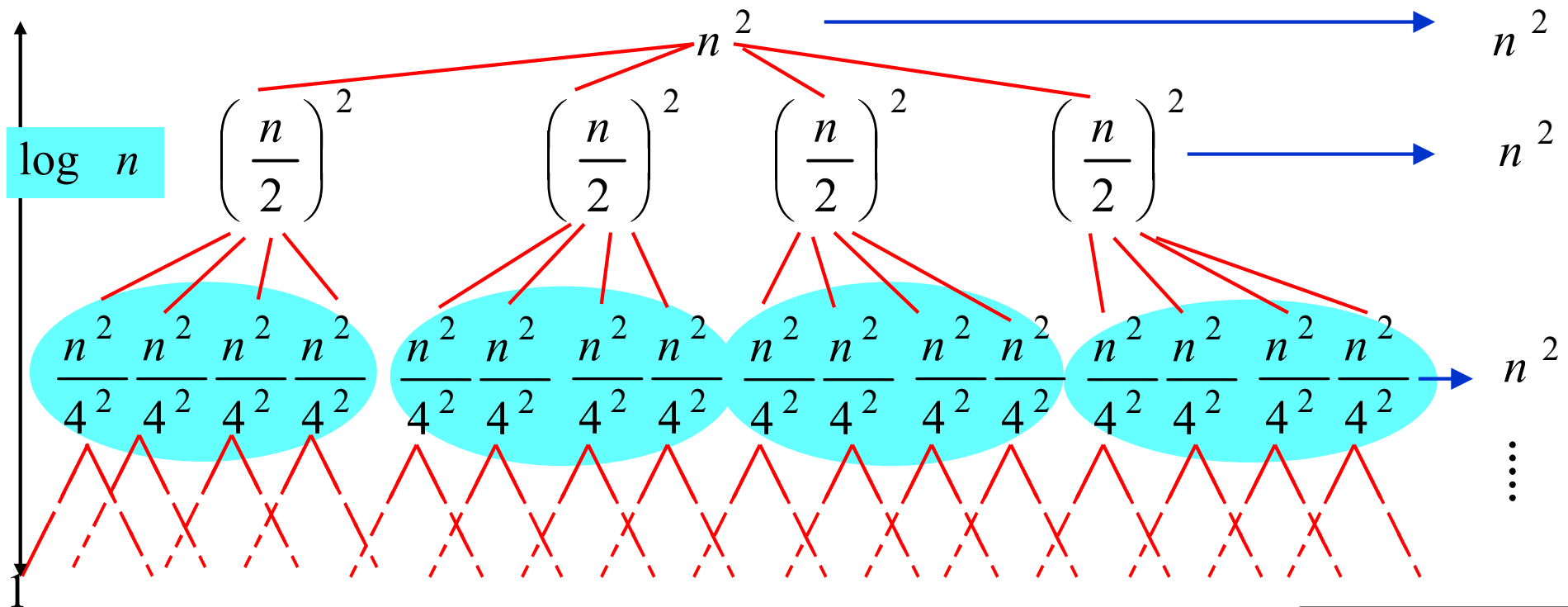
Alberi di Ricorrenza

Esempio: $T(n) = 4T(n/2) + n^2$



Alberi di Ricorrenza

Esempio: $T(n) = 4T(n/2) + n^2$



$$T(n) = \sum_{k=1}^{\log n} n^2 = n^2 \sum_{k=1}^{\log n} 1 = n^2 \log n$$

$$\Theta(n^2 \log n)$$

Analisi di QuickSort: caso medio

Il *tempo di esecuzione* di QuickSort dipende dal *bilanciamento* delle partizioni effettuate dall'algoritmo *Partiziona*

Ci resta da capire come si comporta nel *caso medio*: è più vicino al *caso migliore* o al *caso peggiore*?

Analisi di QuickSort: caso medio

Analizziamo alcuni possibili casi di cattivo bilanciamento delle partizioni.

- ★ Supponiamo che ad ogni chiamata l'algoritmo *Partiziona* produca una partizione che è i **9/10** dell'altra (*partizionamento sbilanciato*)
- 🕒 Supponiamo che ad ogni chiamata l'algoritmo *Partiziona* produca una partizione che è i **99/100** dell'altra (*partizionamento molto sbilanciato*)

Analisi di QuickSort: caso medio

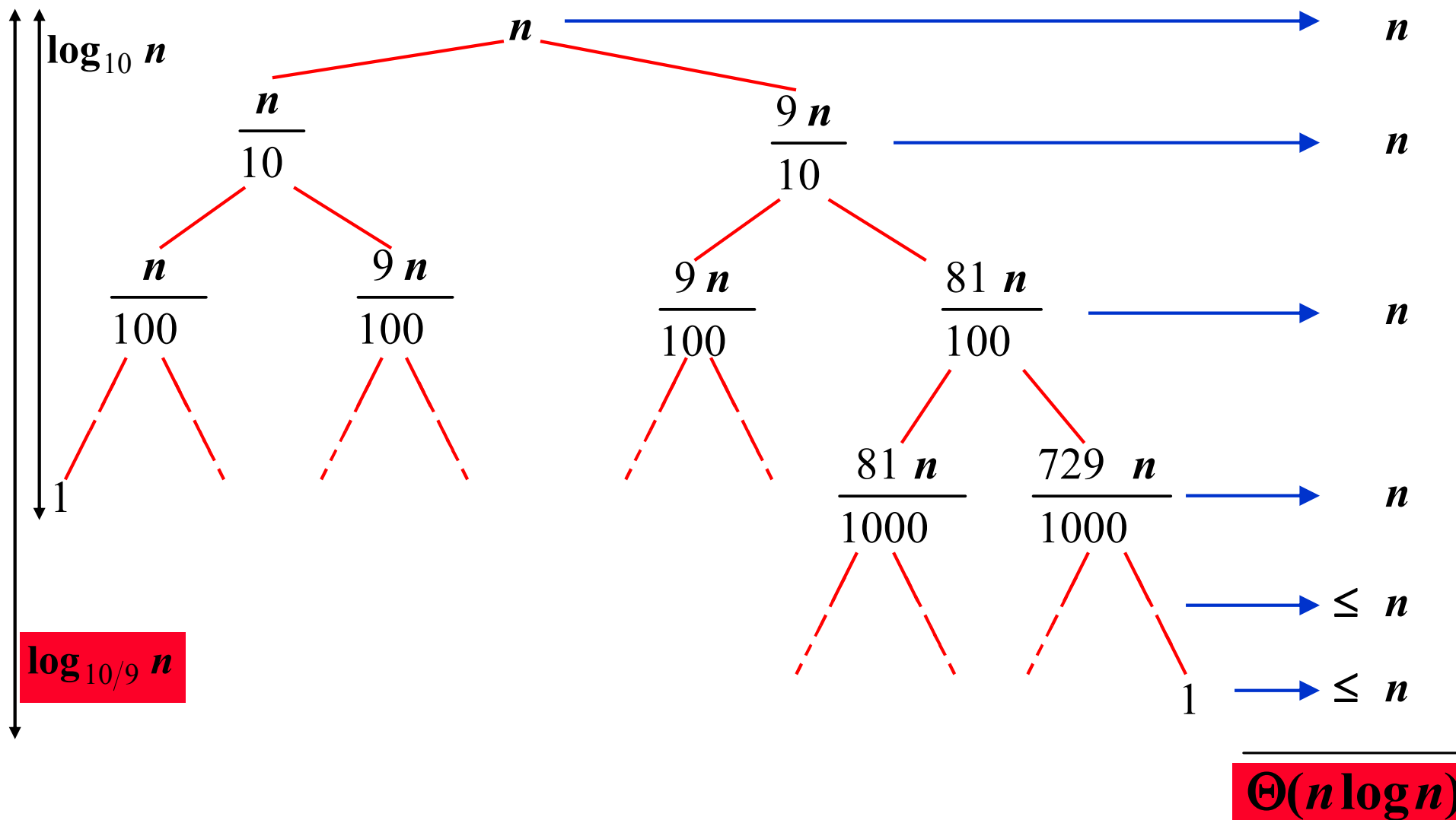
- ☆ Supponiamo che ad ogni chiamata l'algoritmo **Partiziona** produca una partizione che è i **9/10** dell'altra (*partizionamento sbilanciato*)

L'equazione di ricorrenza diventa quindi:

$$T(n) = T(9n/10) + T(n/10) + n$$

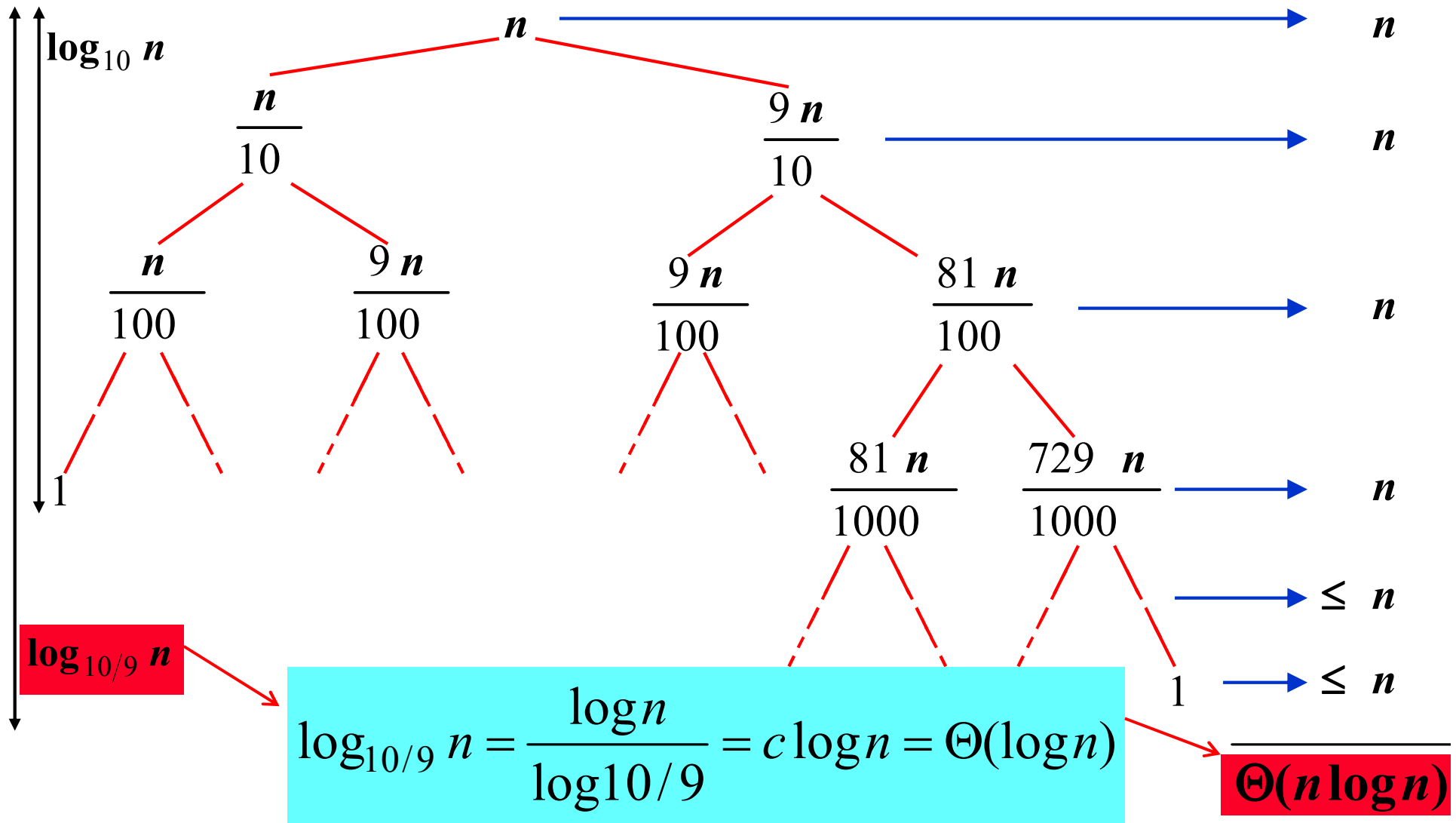
Analisi di QuickSort: caso medio

$$T(n) = T(9n/10) + T(n/10) + n$$



Analisi di QuickSort: caso medio

$$T(n) = T(9n/10) + T(n/10) + n$$



Analisi di QuickSort: caso medio

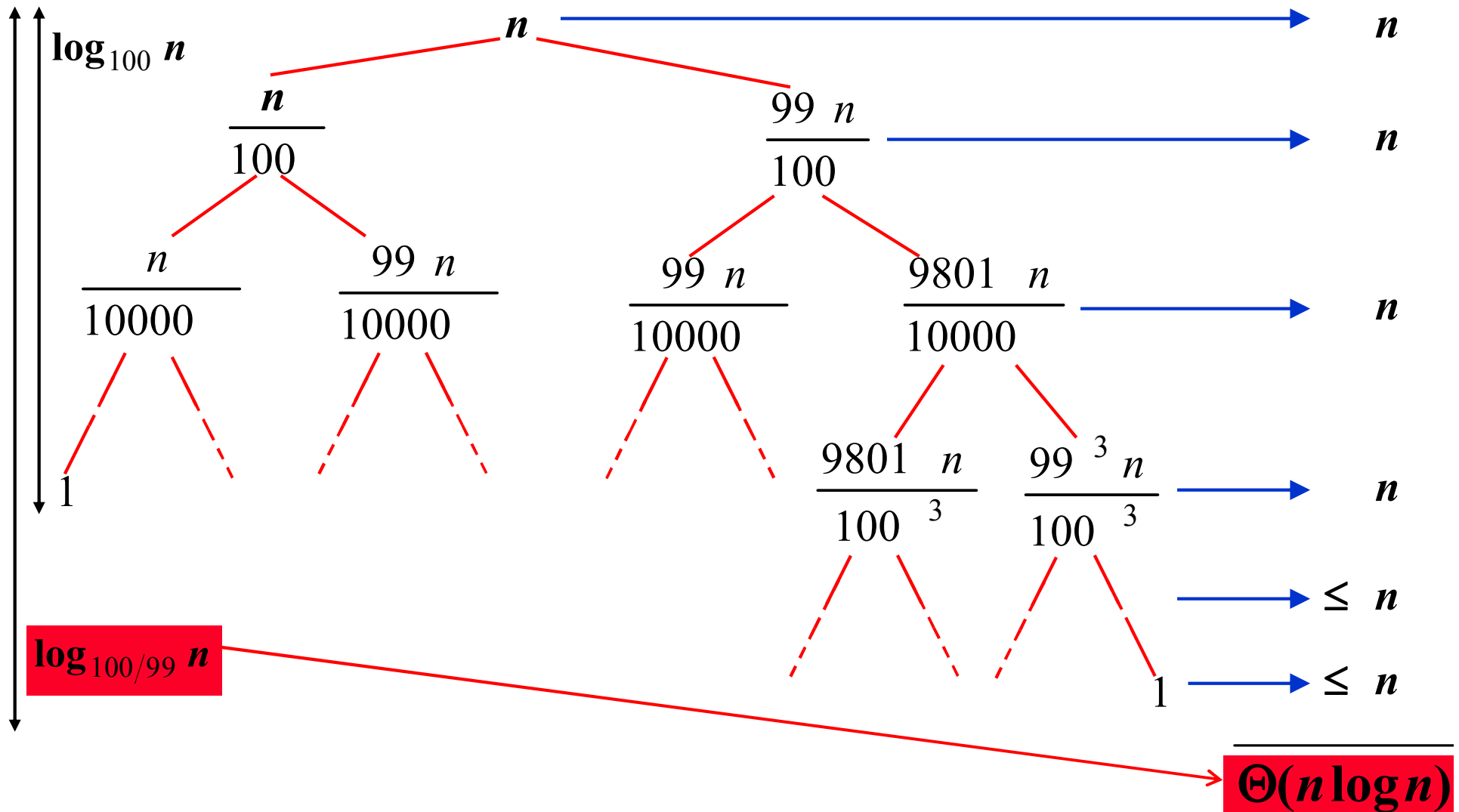
- 🕒 **Supponiamo che ad ogni chiamata l'algoritmo *Partiziona* produca una partizione che è i **99/100** dell'altra (*partizionamento sbilanciato*)**

L'equazione di ricorrenza diventa quindi:

$$T(n) = T(99n/100) + T(n/100) + n$$

Analisi di QuickSort: caso medio

$$T(n) = T(99n/100) + T(n/100) + n$$



Analisi di QuickSort: caso medio

In effetti si può dimostrare che:

ogni volta che **Partiziona** suddivide l'array in **porzioni** che **differiscono** per un **fattore** **proporzionale** **costante**,

il Tempo di Esecuzione è $\Theta(n \log n)$

Analisi di QuickSort: caso medio

È necessario definire una nozione chiara di ***caso medio***.

Assumiamo che tutte le ***permutazioni*** dei valori in input abbiamo ***uguale probabilità***.

Assumendo di eseguire ***QuickSort*** su un array di input casuale (***random***) ci aspettiamo che alcune ***partizioni*** siano ***ben bilanciate*** ed altre ***mal bilanciate***.

Analisi di QuickSort: caso medio

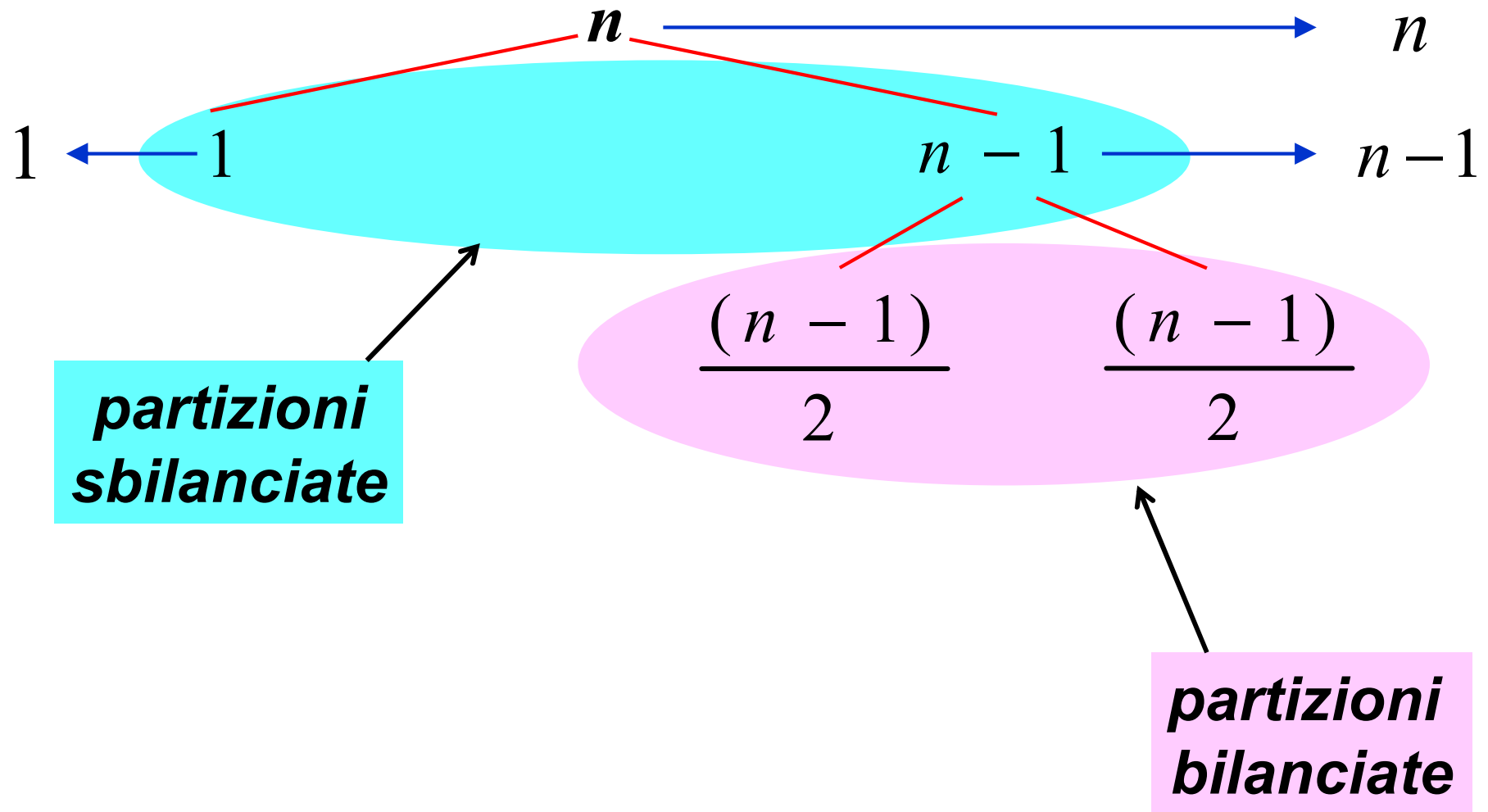
Nel caso medio **Partiziona** produrrà un “*mix*” di *partizioni ben bilanciate* e *mal bilanciate*, distribuite casualmente lungo l'albero di ricorsione.

Supponiamo che le *partizioni ben bilanciate* e quelle *mal bilanciate* si alternino nei diversi livelli dell'albero, cioè:

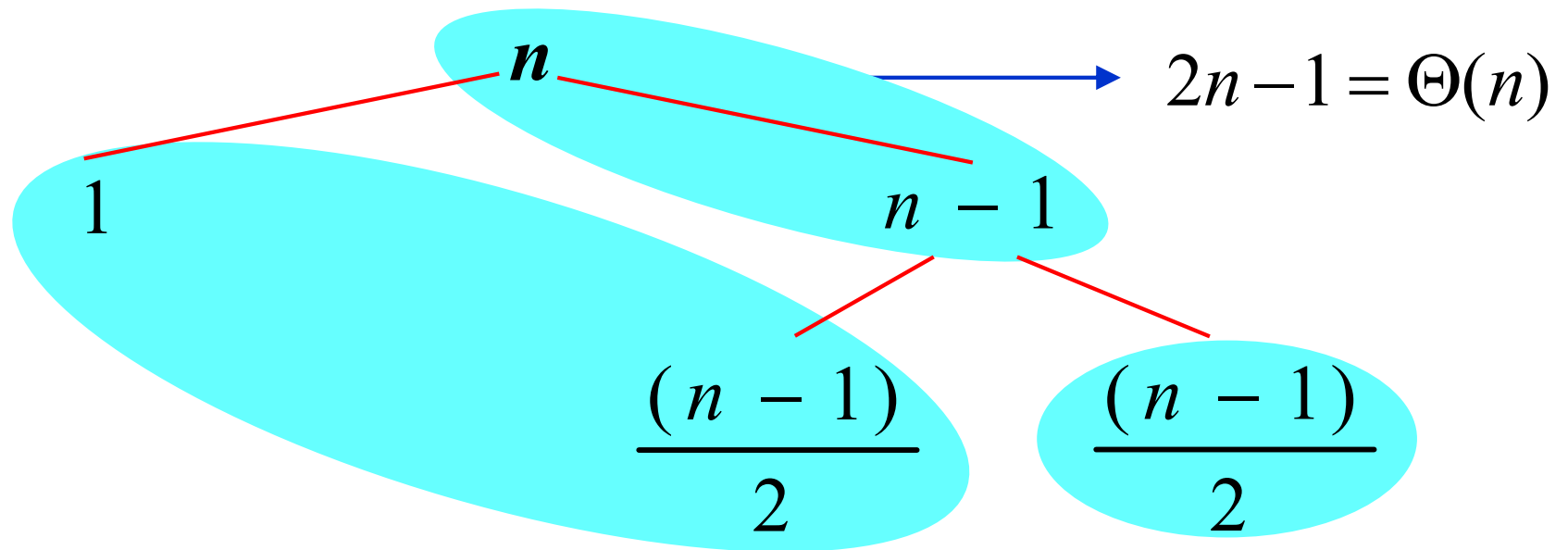
a livello i le partizioni sono di dimensioni 1 e $n-1$

*a livello $i+1$ le partizioni sono di dimensioni $n/2$
ed $n/2$*

Analisi di QuickSort: caso medio

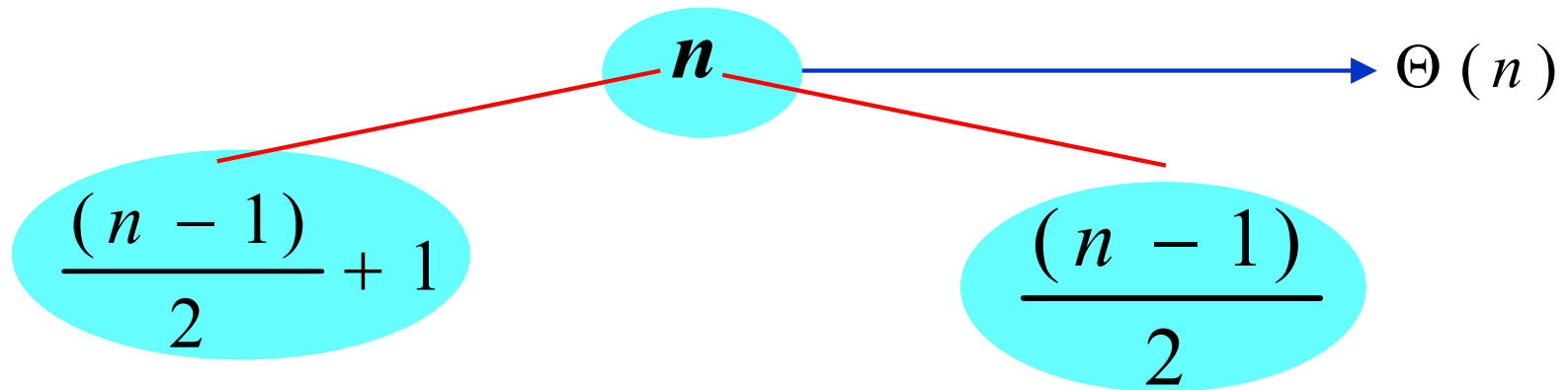


Analisi di QuickSort: caso medio



Combinando il costo di un **partizionamento sbilanciato** seguito da **uno bilanciato**, si ottiene un costo combinato sui due livelli che è $\Theta(n)$

Analisi di QuickSort: caso medio



La situazione del *partizionamento precedente* non è peggiore di questa, che ha ancora un costo dell'ordine di $\Theta(n)$ e rappresenta un *partizionamento piuttosto ben bilanciato*

Analisi di QuickSort: caso medio

*Supponendo allora che le **partizioni ben bilanciate** e quelle **mal bilanciate** si alternino nei diversi livelli dell'albero:*

*otteniamo che in questo caso il **costo medio** è ancora **$O(n \log n)$***

*dove però la notazione **O-grande** nasconde qui una **costante maggiore** che nel **caso migliore***

Analisi di QuickSort

*L'analisi che abbiamo fatto si basa sull'**as-**
sunzione che ciascun input abbia uguale
probabilità di presentarsi.*

*Questa non è però sempre un'assunzione
sufficientemente generale!*

*Possiamo fare di più! Invece di assumere una
distribuzione casuale, è possibile imporla!*

*ad esempio **permutando in maniera casuale**
(alcuni) elementi dell'array in input*

Analisi di QuickSort Random

```
Partiziona-Random(A, p, r)  
  i = Random(p, r)  
  "scambia A[p] con A[i]"  
  return Partiziona(A, p, r)
```

Random(p, r) : ritorna un intero
che è un valore casuale compreso
tra **p** ed **r**.

Analisi di QuickSort Random

```
Partiziona-Random(A, p, r)
  i = Random(p, r)
  "scambia A[p] con A[i]"
  return Partiziona(A, p, r)
```

Sposta in $A[p]$ il valore contenuto in $A[i]$ determinando così una scelta casuale del *Pivot*.

Analisi di QuickSort Random

```
Partiziona-Random (A, p, r)
  i = Random (p, r)
  "scambia A[p] con A[i]"
  return Partiziona (A, p, r)
```

```
Quick-Sort-Random (A, p, r)
  IF p < r
  THEN
    q = Partiziona-Random (A, p, r)
    Quick-Sort-Random (A, p, q)
    Quick-Sort-Random (A, q + 1, r)
```

Analisi di QuickSort Random

La versione casuale di QuickSort presentata:

non modifica le prestazioni nel caso peggiore (che rimane quadratico) Perché?

ma rende le prestazioni indipendenti dall'ordinamento iniziale dell'array di input

non c'è alcun particolare input che determina il verificarsi del caso peggiore.

Analisi di QuickSort Random: Caso Peggior

Partiziona *suddivide un array di dimensione n in due partizioni di dimensioni che diremo q e $n - q$, rispettivamente.*

Per calcolare il caso peggiore, cercheremo di calcolare il valore massimo del tempo di esecuzione dato dalla ricorrenza

$$T(n) = T(q) + T(n - q) + \Theta(n)$$

Analisi di QuickSort Random: Caso Peggior

Partiziona suddivide un array di dimensione n in due partizioni di dimensioni che diremo q e $n - q$, rispettivamente.

Per calcolare il caso peggiore, cercheremo di calcolare il valore **massimo**, al variare di q , del tempo di esecuzione dato dalla ricorrenza

$$T(n) = T(q) + T(n - q) + \Theta(n)$$

Cioè:

$$T(n) = \max_{1 \leq q \leq n-1} \{T(q) + T(n - q)\} + \Theta(n)$$

Analisi di QuickSort Random: Caso Peggior

$$T(n) = \max_{1 \leq q \leq n-1} \{ T(q) + T(n-q) \} + \Theta(n)$$

Usiamo il metodo di sostituzione

Ipotizziamo $T(n) \leq cn^2$

Sostituendo otteniamo

$$\begin{aligned} T(n) &\leq \max_{1 \leq q \leq n-1} \{ cq^2 + c(n-q)^2 \} + \Theta(n) \\ &\leq c \max_{1 \leq q \leq n-1} \{ q^2 + (n-q)^2 \} + \Theta(n) \end{aligned}$$

Analisi di QuickSort Random: Caso Peggior

$$T(n) = \max_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$T(n) \leq c \max_{1 \leq q \leq n-1} \{q^2 + (n-q)^2\} + \Theta(n)$$

Ci serve sapere quando $q^2 + (n-q)^2$ raggiunge il valore massimo tra 1 e $n-1$

Calcoliamo la sua derivata prima:

$$2q - 2(n-q) = 4q - 2n$$

che è negativa per $q < n/2$ e positiva per $q > n/2$

Analisi di QuickSort Random: Caso Peggior

$$T(n) = \max_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$T(n) \leq c \max_{1 \leq q \leq n-1} \{q^2 + (n-q)^2\} + \Theta(n)$$

La derivata prima:

$$2q - 2(n-q) = 4q - 2n$$

è negativa per $q < n/2$ e positiva per $q > n/2$

Quindi, $q^2 + (n-q)^2$ nell'intervallo $[1, n-1]$ raggiunge il valore massimo quando $q=1$ o $q=n-1$.

Analisi di QuickSort Random: Caso Peggior

$$T(n) = \max_{1 \leq q \leq n-1} \{ T(q) + T(n-q) \} + \Theta(n)$$

$$\begin{aligned} T(n) &\leq c \max_{1 \leq q \leq n-1} \{ q^2 + (n-q)^2 \} + \Theta(n) \\ &\leq c (1^2 + (n-1)^2) + \Theta(n) \\ &\leq c (n^2 - 2(n-1)) + \Theta(n) \\ &\leq c n^2 - 2c (n-1) + \Theta(n) \end{aligned}$$

Analisi di QuickSort Random: Caso Peggior

$$T(n) = \max_{1 \leq q \leq n-1} \{ T(q) + T(n-q) \} + \Theta(n)$$

$$\begin{aligned} T(n) &\leq c \max_{1 \leq q \leq n-1} \{ q^2 + (n-q)^2 \} + \Theta(n) \\ &\leq c (1^2 + (n-1)^2) + \Theta(n) \\ &\leq c (n^2 - 2(n-1)) + \Theta(n) \\ &\leq c n^2 - 2c (n-1) + \Theta(n) \\ &\leq c n^2 \end{aligned}$$

poiché possiamo scegliere c abbastanza grande da rendere $2c (n-1)$ dominante su $\Theta(n)$

Analisi di QuickSort Random: Caso Migliore

Partiziona suddivide un array di dimensione n in due partizioni di dimensioni che diremo q e $n - q$, rispettivamente.

Per calcolare il caso migliore, cercheremo di calcolare il valore **minimo** del tempo di esecuzione dato dalla ricorrenza

$$T(n) = T(q) + T(n - q) + \Theta(n)$$

Cioè:

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n - q)\} + \Theta(n)$$

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

Usiamo il metodo di sostituzione

Ipotizziamo $T(n) \leq c n \log n$

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{ T(q) + T(n-q) \} + \Theta(n)$$

Usiamo il metodo di sostituzione

Ipotizziamo $T(n) \leq c n \log n$

Sostituendo otteniamo

$$\begin{aligned} T(n) &\leq \min_{1 \leq q \leq n-1} \{ c q \log q + c (n-q) \log (n-q) \} + \Theta(n) \\ &\leq c \min_{1 \leq q \leq n-1} \{ q \log q + (n-q) \log (n-q) \} + \Theta(n) \end{aligned}$$

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$T(n) \leq c \min_{1 \leq q \leq n-1} \{q \log q + (n-q) \log (n-q)\} + \Theta(n)$$

Ci serve sapere quando $q \log q + (n-q) \log (n-q)$ raggiunge il valore minimo tra 1 e $n-1$

Calcoliamo la sua derivata prima:

$$\log q - \log(n-q)$$

che è nulla per $q = n/2$, negativa per $q < n/2$ e positiva per $q > n/2$ (quindi $q = n/2$ è un minimo)

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$T(n) \leq c \min_{1 \leq q \leq n-1} \{q \log q + (n-q) \log (n-q)\} + \Theta(n)$$

La derivata prima:

$$\log q - \log(n-q)$$

che è nulla per $q = n/2$, negativa per $q < n/2$ e positiva per $q > n/2$ (cioè $q = n/2$ è un minimo)

Quindi $q \log q + (n-q) \log (n-q)$ raggiunge il valore minimo tra 1 e $n-1$ quando $q = n/2$

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$\begin{aligned} T(n) &\leq c \min_{1 \leq q \leq n-1} \{q \log q + (n-q) \log (n-q)\} + \Theta(n) \\ &\leq c (n \log n/2) + \Theta(n) \\ &\leq c n \log n - c n + \Theta(n) \\ &\leq c n \log n - c n + \Theta(n) \end{aligned}$$

Analisi di QuickSort Random: Caso Migliore

$$T(n) = \min_{1 \leq q \leq n-1} \{T(q) + T(n-q)\} + \Theta(n)$$

$$\begin{aligned} T(n) &\leq c \min_{1 \leq q \leq n-1} \{q \log q + (n-q) \log (n-q)\} + \Theta(n) \\ &\leq c (n \log n/2) + \Theta(n) \\ &\leq c n \log n - c n + \Theta(n) \\ &\leq c n \log n - c n + \Theta(n) \\ &\leq c n \log n \end{aligned}$$

poiché possiamo scegliere c abbastanza grande da rendere $c n$ dominante su $\Theta(n)$

Analisi di QuickSort Random: Caso Medio

*Quello che dobbiamo fare è costruire l'**equazione di ricorrenza** per il caso medio.*

⑤ *Assumeremo che **tutti** gli **elementi** siano **distinti** per semplificare l'analisi.*

⑤ ***Partiziona-Random** chiama **Partiziona** dopo aver scambiato **A[p]** con un elemento a caso dell'array*

⑤ *quale sarà allora il valore di **q** ritornato da **Partiziona**?*

Analisi di QuickSort Random: Caso Medio

*Quale sarà allora il valore di q ritornato
Partiziona?*

- ⑤ *Dipenderà dal **rango** di $A[p]$ (che è un elemento casuale dell'array).*
- ⑤ *Il **rango** di un numero x rispetto a $A[p, \dots, r]$ è il numero di elementi di $A[p, \dots, r]$ che sono **minori o uguali** ad x*

Analisi di QuickSort Random: Caso Medio

*Quale sarà allora il valore di q ritornato
Partiziona?*

⑤ *Dipenderà dal **rango** di $A[p]$ (che è un
elemento casuale dell'array).*

⑤ *Essendo $A[p]$ un elemento casuale
dell'array, la **probabilità** che il **rango** di $A[p]$
sia i (con $i = 1, \dots, n$) sarà $1/n$ (dove $n = r - p + 1$)*

*poiché tutti gli elementi hanno uguale probabilità di
essere scelti e sono tutti distinti.*

Analisi di QuickSort Random: Caso Medio

Quale sarà allora il valore di q ritornato
Partiziona?

- ⑤ Se il **rango** è **1** Partiziona ritornerà una partizione lunga **1** e una lunga **$n-1$**
- ⑤ Se il **rango** è **2** Partiziona ritornerà ancora una partizione lunga **1** e una lunga **$n-1$**
- ⑤ ...
- ⑤ Se il **rango** è **h** Partiziona ritornerà una partizione lunga **$h-1$** e una lunga **$n-h+1$**
- ⑤ Se il **rango** è **n** Partiziona ritornerà una partizione lunga **$n-1$** e una lunga **1**

Analisi di QuickSort Random: Caso Medio

*Quale sarà allora il valore di q ritornato
Partiziona?*

⑤ *Se il rango è 1 Partiziona ritornerà una
partizione lunga 1 e una lunga $n-1$*

⑤ *Se il rango è h (per $h \geq 2$) Partiziona
ritornerà una partizione lunga $h-1$ e una lunga
 $n-h+1$*

ciascun caso ha probabilità $1/n$

Analisi di QuickSort Random: Caso Medio

*Quale sarà allora il valore di q ritornato
Partiziona?*

⑤ *Se il rango è 1 Partiziona ritornerà una
partizione lunga 1 e una lunga $n-1$*

*allora $q = 1$ e QuickSort sarà chiamato
ricorsivamente su partizioni di dimensioni 1 e
 $n-1$*

con probabilità $1/n$

Analisi di QuickSort Random: Caso Medio

*Quale sarà allora il valore di q ritornato
Partiziona?*

⊕ *Se il rango è h (per $h \geq 2$) Partiziona
ritornerà una partizione lunga $h-1$ e una lunga
 $n-h+1$*

*allora $q = h - 1$ e QuickSort sarà chiamato
ricorsivamente su partizioni di dimensioni $h-1$
e $n-h+1$*

con probabilità $1/n$

Analisi di QuickSort Random: Caso Medio

$$T(n) = \left[\text{grey bar} \right] \left(T(1) + T(n-1) \right) + \left[\text{pink bar} \right] + \Theta(n)$$

Se il **rango** è **1** **Partiziona**
ritornerà una partizione
lunga **1** e una lunga **n-1**

Analisi di QuickSort Random: Caso Medio

$$T(n) = \left(T(1) + T(n-1) + \sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

Se il **rango** è **1** **Partiziona** ritornerà una partizione lunga **1** e una lunga **n-1**

Se il **rango** è **h** (per $h \geq 2$) **Partiziona** ritornerà una partizione lunga **h-1** e una lunga **n-h+1** (q varia tra **1** e **n-1**)

Analisi di QuickSort Random: Caso Medio

$$T(n) = \frac{1}{n} \left(T(1) + T(n-1) + \sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

Se il **rango** è **1** **Partiziona**
ritornerà una partizione
lunga **1** e una lunga **n-1**

ciascun caso ha
probabilità **1/n**

Se il **rango** è **h** (per **h ≥ 2**)
Partiziona ritornerà una
partizione lunga **h-1** e una lunga
n-h+1 (**q** varia tra **1** e **n-1**)

Analisi di QuickSort Random: Caso Medio

L'equazione di ricorrenza per il caso medio sarà quindi:

$$T(n) = \frac{1}{n} \left(T(1) + T(n-1) + \sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

Analisi di QuickSort Random: Caso Medio

$$T(n) = \frac{1}{n} \left(T(1) + T(n-1) + \sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

$$\frac{1}{n} (T(1) + T(n-1)) = \frac{1}{n} (\Theta(1) + O(n^2))$$

Analisi di QuickSort Random: Caso Medio

$$T(n) = \frac{1}{n} \left(T(1) + T(n-1) + \sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

$$\frac{1}{n} (T(1) + T(n-1)) = \frac{1}{n} (\Theta(1) + O(n^2))$$

$$= \frac{1}{n} O(n^2) = O(n)$$

Analisi di QuickSort Random: Caso Medio

$$T(n) = \frac{1}{n} \left(\sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

poiché $O(n)$ viene assorbito da $\Theta(n)$

$$\frac{1}{n} (T(1) + T(n-1)) = O(n)$$

Analisi di QuickSort Random: Caso Medio

$$T(n) = \frac{1}{n} \left(\sum_{q=1}^{n-1} (T(q) + T(n-q)) \right) + \Theta(n)$$

$$= \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

poiché per q che varia fra 1 e $n-1$ ciascun valore di $T(q)$ compare due volte nella sommatoria, una volta come $T(q)$ ed una come $T(n-q)$.

Analisi di QuickSort Random: Caso Medio

L'equazione di ricorrenza diviene:

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

La risolveremo col metodo di sostituzione

Analisi di QuickSort Random: Caso Medio

L'equazione di ricorrenza diviene:

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

Vogliamo dimostrare che $T(n) = O(n \log n)$

Analisi di QuickSort Random: Caso Medio

L'equazione di ricorrenza diviene:

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

Ipotizziamo

$$T(n) \leq a n \log n$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2}{n} \left(\sum_{q=1}^{n-1} aq \log q \right) + \Theta(n)$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2}{n} \left(\sum_{q=1}^{n-1} a q \log q \right) + \Theta(n)$$

$$\leq \frac{2a}{n} \sum_{q=1}^{n-1} q \log q + \Theta(n)$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2a}{n} \sum_{q=1}^{n-1} q \log q + \Theta(n)$$

poiché si può dimostrare che

$$\sum_{q=1}^{n-1} q \log q \leq \frac{1}{2} n^2 \log n - \frac{1}{8} n^2$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2a}{n} \sum_{q=1}^{n-1} q \log q + \Theta(n)$$

$$\leq \frac{2a}{n} \left(\frac{1}{2} n^2 \log n - \frac{1}{8} n^2 \right) + \Theta(n)$$

$$\sum_{q=1}^{n-1} q \log q \leq \frac{1}{2} n^2 \log n - \frac{1}{8} n^2$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2a}{n} \left(\frac{1}{2} n^2 \log n - \frac{1}{8} n^2 \right) + \Theta(n)$$

$$\leq an \log n - \frac{a}{4} n + \Theta(n)$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq \frac{2a}{n} \left(\frac{1}{2} n^2 \log n - \frac{1}{8} n^2 \right) + \frac{2b}{n} (n-1) + \Theta(n)$$

$$\leq an \log n - \frac{a}{4} n + 2b + \Theta(n)$$

$$\leq an \log n + \left(\Theta(n) - \frac{a}{4} n \right)$$

Analisi di QuickSort Random: Caso Medio

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

$$T(n) = \frac{2}{n} \left(\sum_{q=1}^{n-1} T(q) \right) + \Theta(n)$$

$$\leq an \log n - \frac{a}{4} n + \Theta(n)$$

$$\leq an \log n + \left(\Theta(n) - \frac{a}{4} n \right)$$

Scegliendo a grande abbastanza da rendere $a n/4$ dominante su $\Theta(n)$

$$\leq an \log n$$

Analisi di QuickSort Random: Caso Medio

Possiamo concludere che

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

A patto di dimostrare che

$$\sum_{q=1}^{n-1} q \log q \leq \frac{1}{2} n^2 \log n - \frac{1}{8} n^2$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k \leq \log n \sum_{k=1}^{n-1} k$$

$$\leq \frac{1}{2} n(n-1) \log n = \frac{n^2 - n}{2}$$

$$\leq n^2 \log n$$

Questo limite non è però sufficiente per risolvere la ricorrenza, ma quello che abbiamo calcolato sarà utile per trovare uno adeguato!

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k = \sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k + \sum_{k=\lceil n/2 \rceil}^{n-1} k \log k$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k \leq \log(n/2) \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=1}^{n-1} k \log k \leq \log n \sum_{k=1}^{n-1} k$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k = \sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k + \sum_{k=\lceil n/2 \rceil}^{n-1} k \log k$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=1}^{n-1} k \log k \leq \log n \sum_{k=1}^{n-1} k$$

$$\sum_{k=\lceil n/2 \rceil}^{n-1} k \log k \leq \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k = \sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k + \sum_{k=\lceil n/2 \rceil}^{n-1} k \log k$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=\lceil n/2 \rceil}^{n-1} k \log k \leq \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=\lceil n/2 \rceil}^{n-1} k \log k \leq \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

Analisi di QuickSort Random: Caso Medio

$$\begin{aligned} \sum_{k=1}^{n-1} k \log k &\leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k \\ &\leq \log n \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k \end{aligned}$$

Analisi di QuickSort Random: Caso Medio

$$\begin{aligned} \sum_{k=1}^{n-1} k \log k &\leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k \\ &\leq \log n \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k \\ &\leq \log n \sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k \end{aligned}$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

$$\leq \log n \sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=1}^{n-1} k = \frac{1}{2} n(n-1)$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k = \frac{1}{2} \frac{n}{2} \left(\frac{n}{2} - 1 \right)$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

$$\leq \log n \sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=1}^{n-1} k = \frac{1}{2} n(n-1)$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k = \frac{1}{2} \frac{n}{2} \left(\frac{n}{2} - 1 \right)$$

$$\leq \frac{1}{2} n(n-1) \log n - \frac{1}{2} \frac{n}{2} \left(\frac{n}{2} - 1 \right)$$

Analisi di QuickSort Random: Caso Medio

$$\sum_{k=1}^{n-1} k \log k \leq (\log n - 1) \sum_{k=1}^{\lceil n/2 \rceil - 1} k + \log n \sum_{k=\lceil n/2 \rceil}^{n-1} k$$

$$\leq \log n \sum_{k=1}^{n-1} k - \sum_{k=1}^{\lceil n/2 \rceil - 1} k$$

$$\sum_{k=1}^{n-1} k = \frac{1}{2} n(n-1)$$

$$\sum_{k=1}^{\lceil n/2 \rceil - 1} k = \frac{1}{2} \frac{n}{2} \left(\frac{n}{2} - 1 \right)$$

$$\leq \frac{1}{2} n(n-1) \log n - \frac{1}{2} \frac{n}{2} \left(\frac{n}{2} - 1 \right)$$

$$\leq \frac{1}{2} n^2 \log n - \frac{1}{8} n^2$$

Analisi di QuickSort Random: Caso Medio

Possiamo concludere che:

↪ **nel caso medio, il tempo di esecuzione è:**

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

✦ **nel caso migliore, il tempo di esecuzione è:**

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

✦ **nel caso peggiore, il tempo di esecuzione è:**

$$T(n) = O(n^2)$$