# *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*

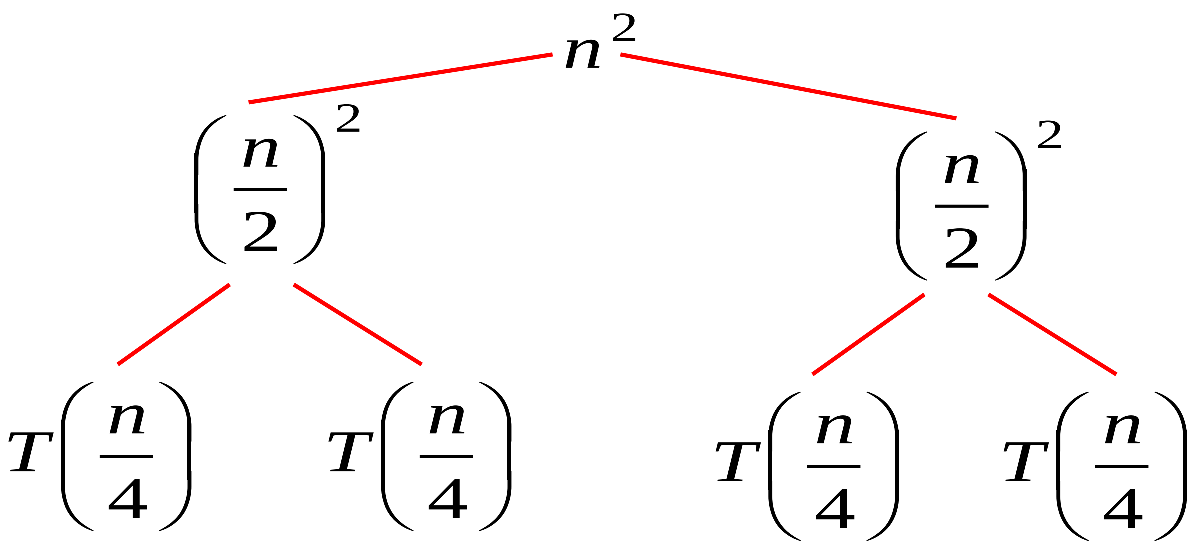
***Esempio:*** $T(n) = 2T(n/2) + n^2$

$$n^2$$

$$T\left(\frac{n}{2}\right) \qquad\qquad T\left(\frac{n}{2}\right)$$

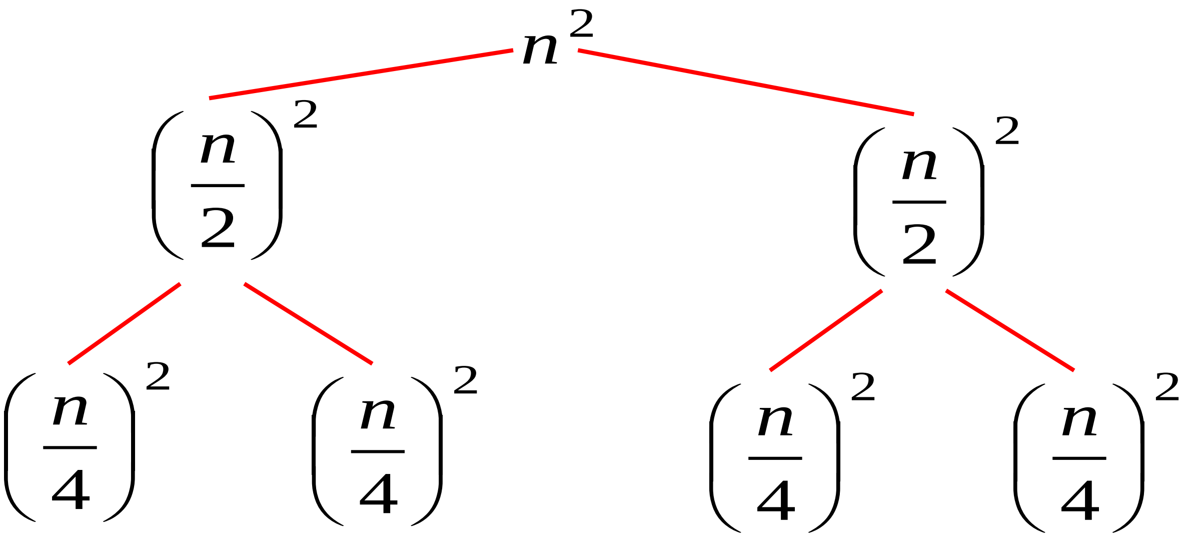
# Alberi di Ricorrenza

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



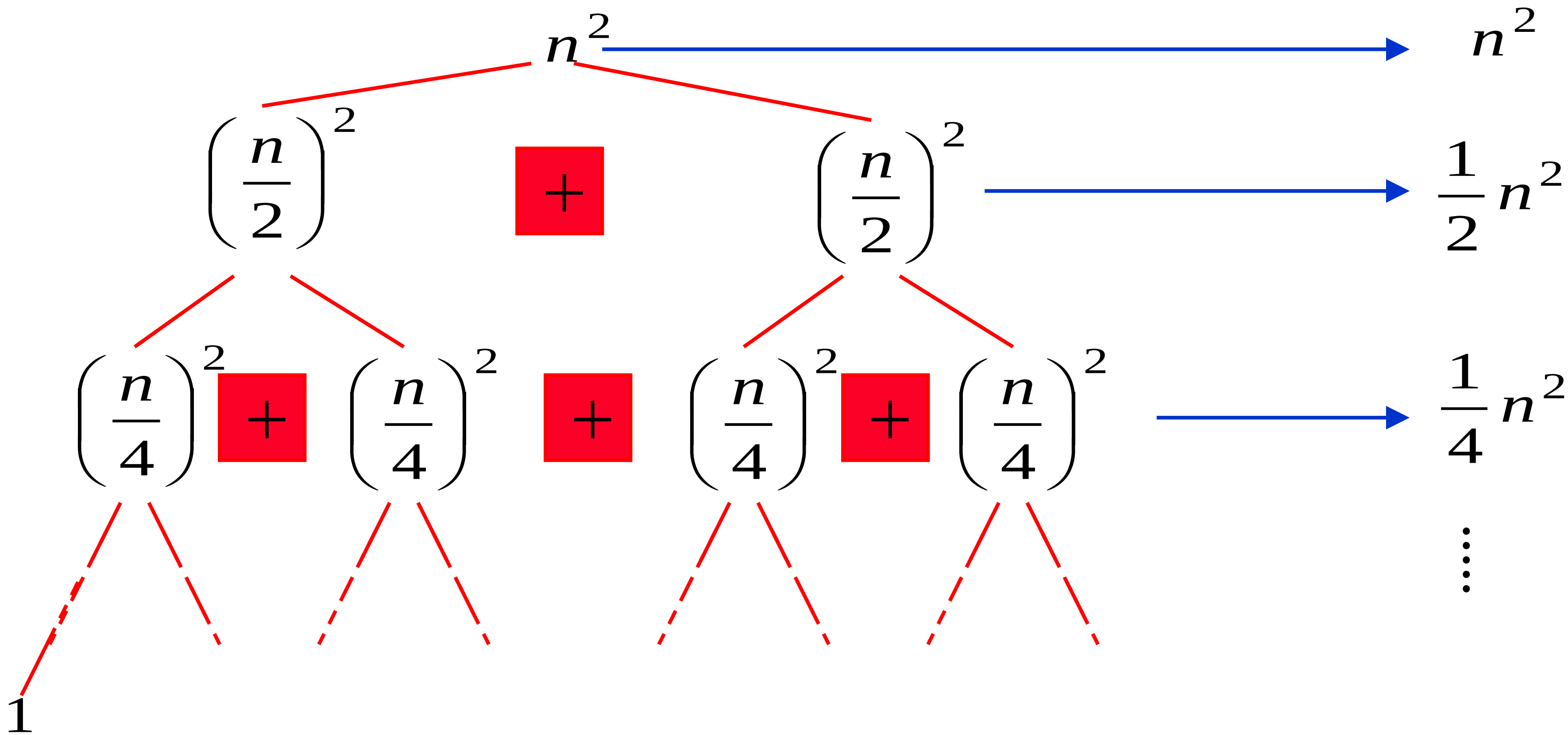
# *Alberi di Ricorrenza*

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



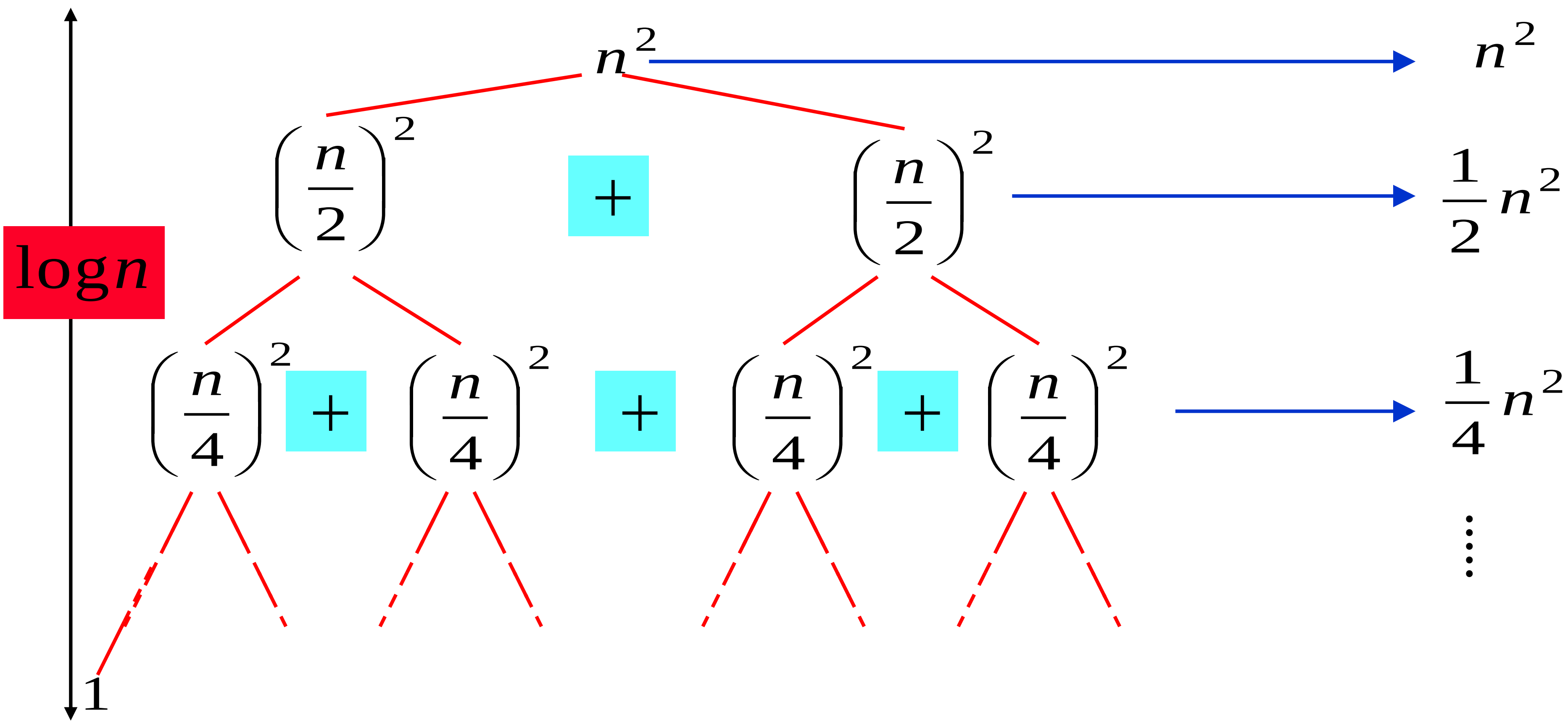
# Alberi di Ricorrenza

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



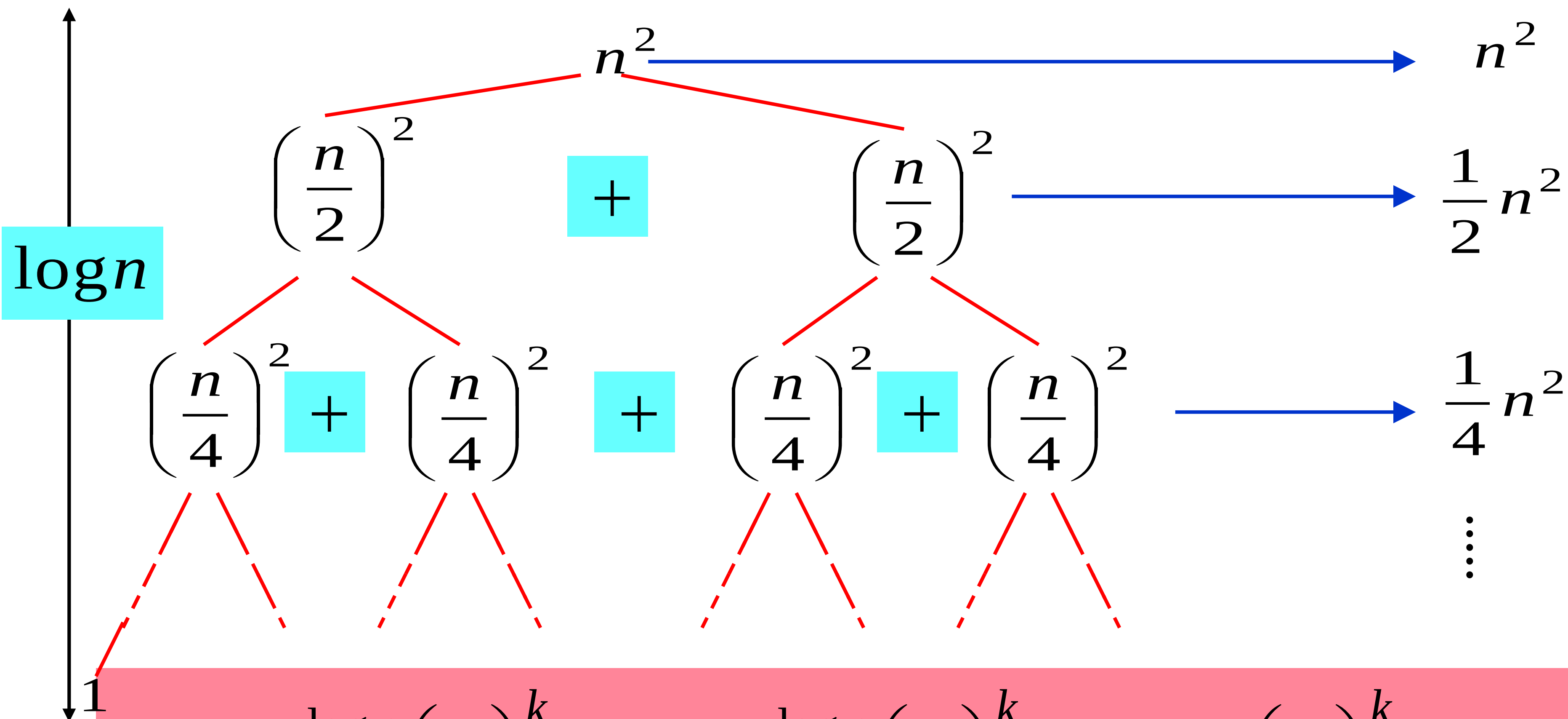
# *Alberi di Ricorrenza*

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



# Alberi di Ricorrenza

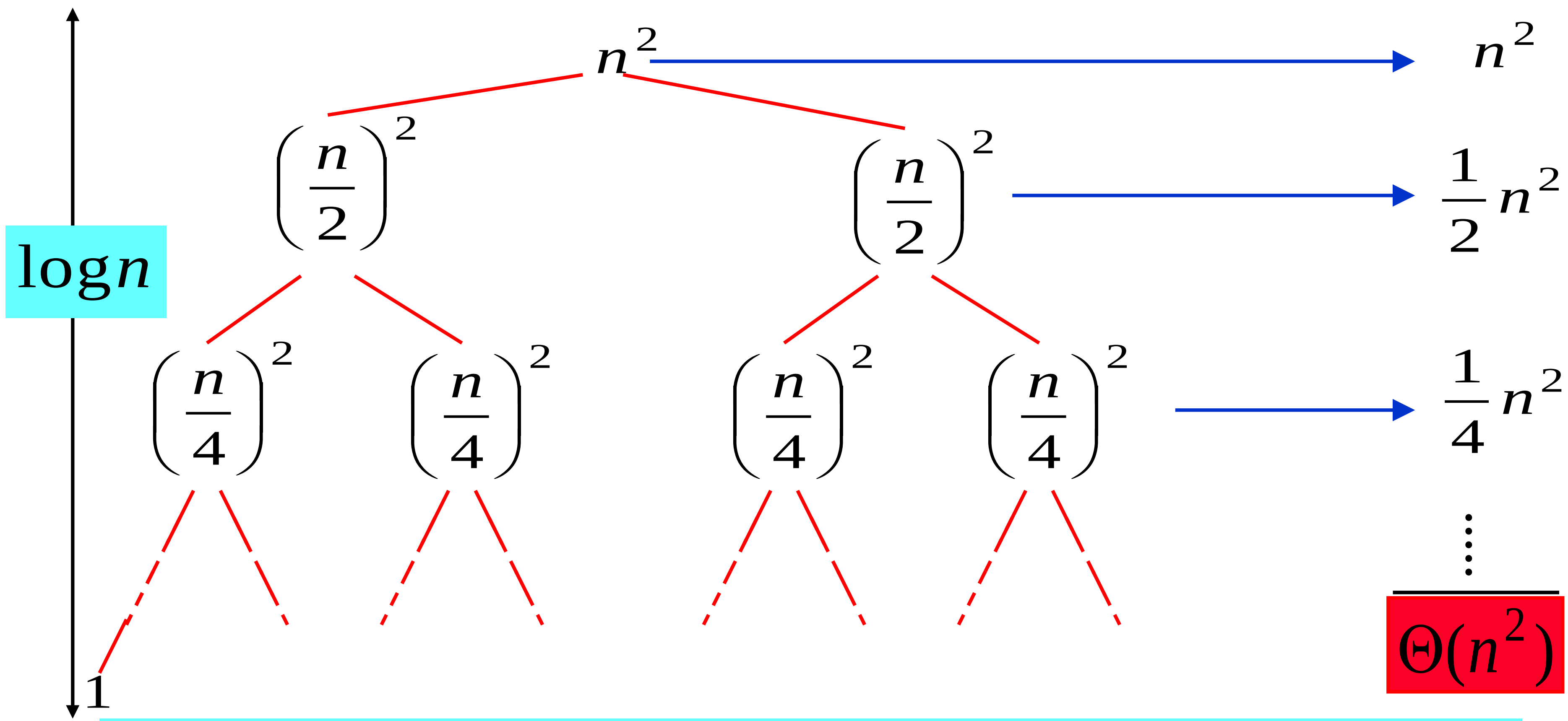
**Esempio:** $T(n) = 2T(n/2) + n^2$




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

# Alberi di Ricorrenza

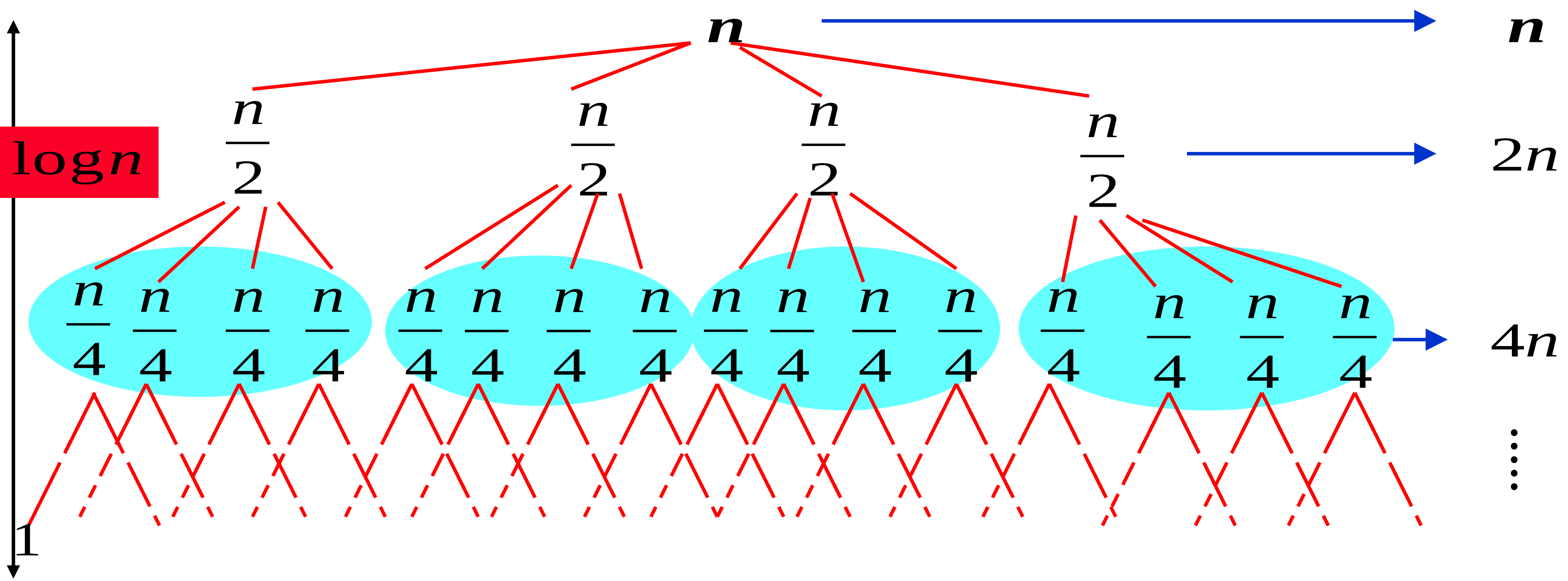
**Esempio:** $T(n) = 2T(n/2) + n^2$




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

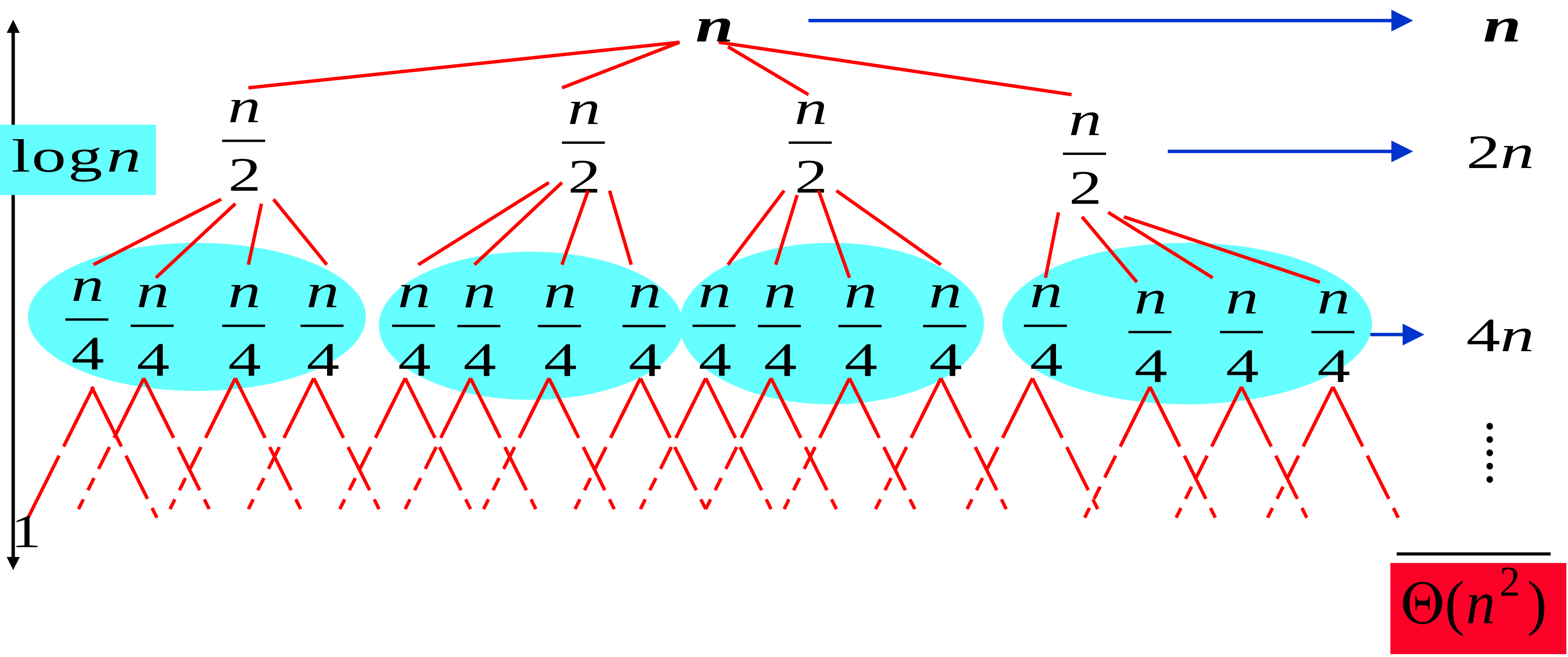
# Alberi di Ricorrenza

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



# Alberi di Ricorrenza

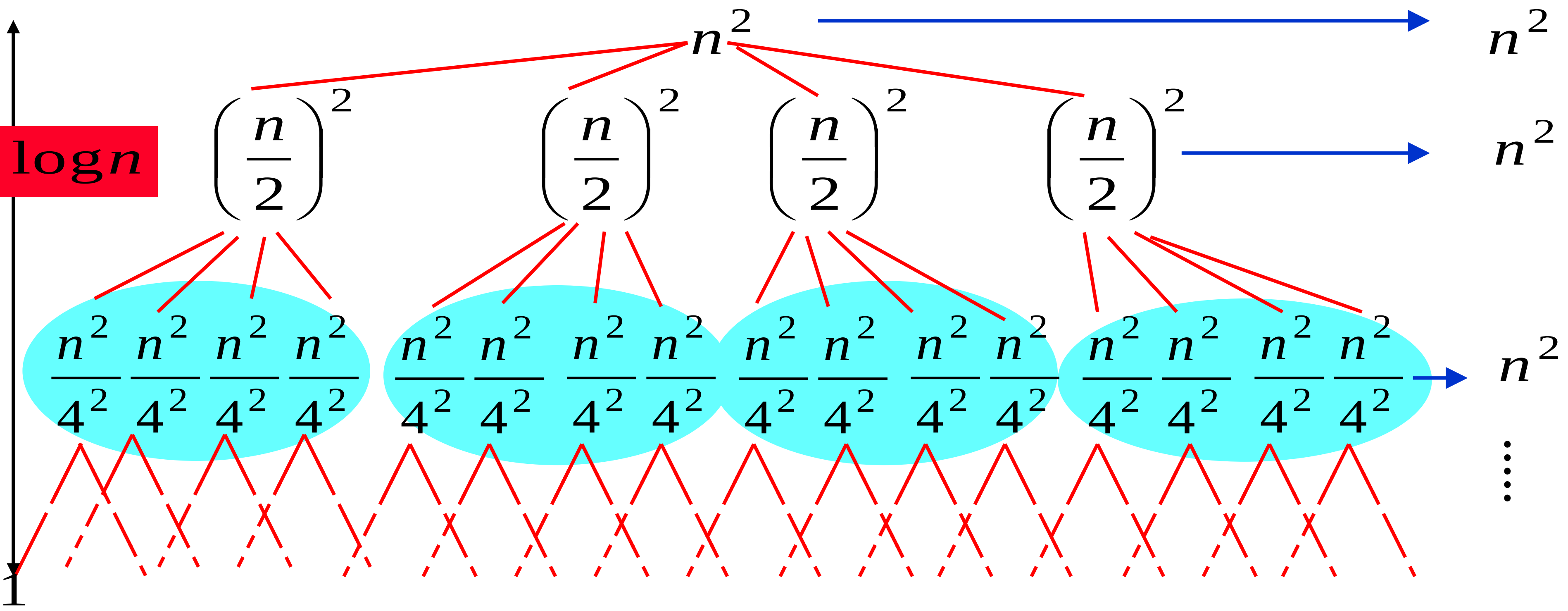
**Esempio:** $T(n) = 4T(n/2) + n$




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

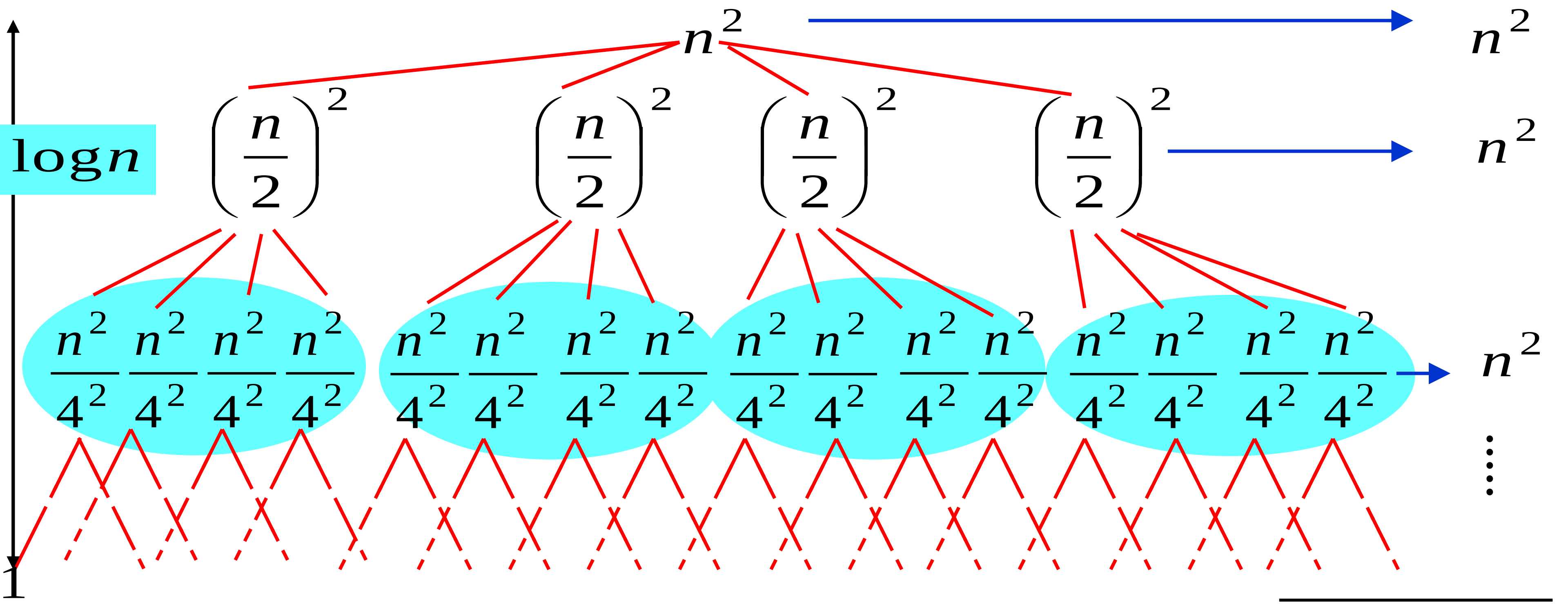
# Alberi di Ricorrenza

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



# Alberi di Ricorrenza

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




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

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

# *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 9 volte l'altra (*partizionamento sbilanciato*)**

- **Supponiamo che ad ogni chiamata l'algoritmo `Partiziona` produca una partizione che è 99 volte l'altra (*partizionamento molto sbilanciato*)**

# *Analisi di QuickSort: caso medio*

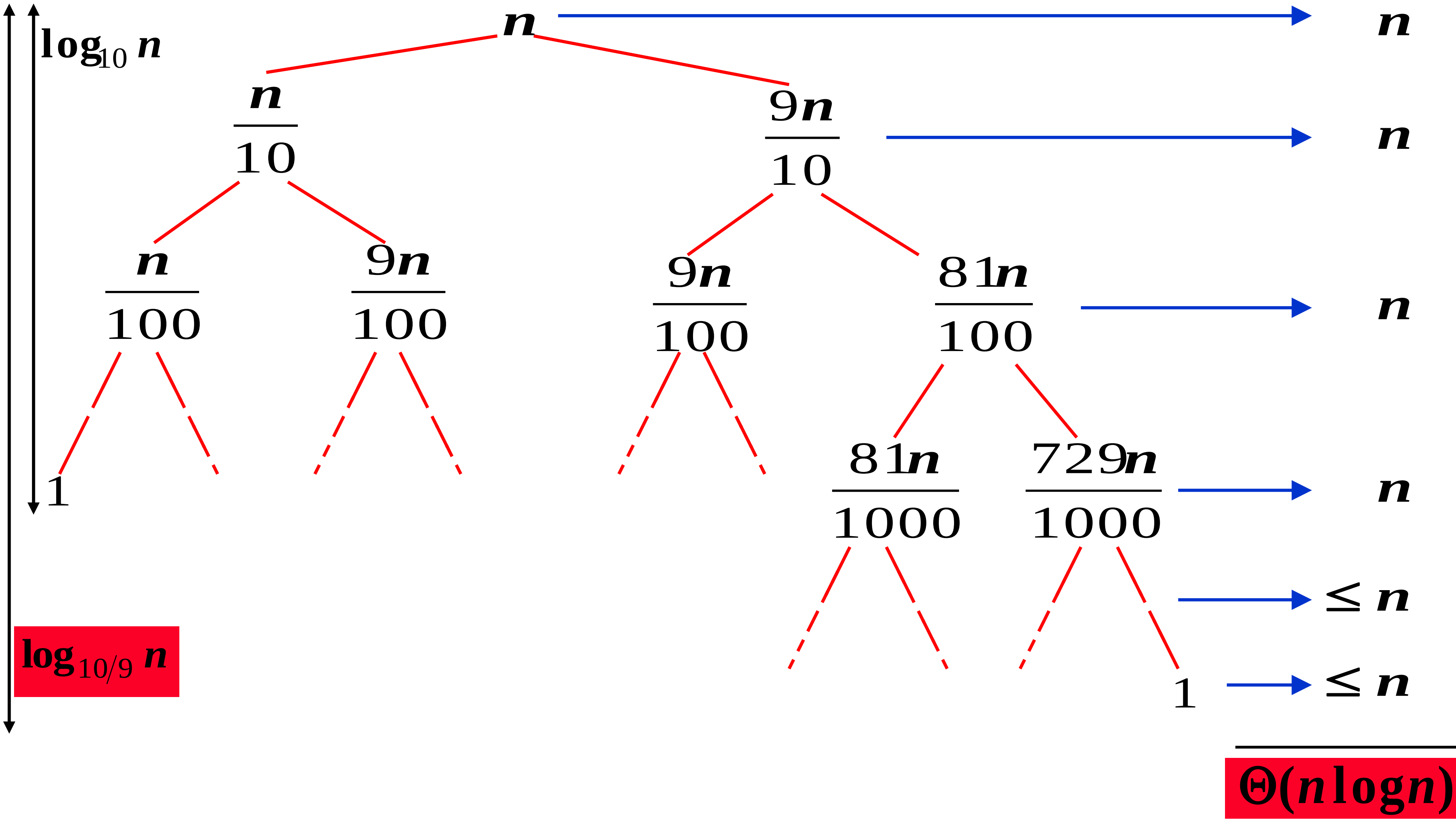
- **Supponiamo che ad ogni chiamata l'algoritmo `Partiziona` produca una partizione che è 9 volte l'altra (*partizionamento sbilanciato*)**

**L'equazione di ricorrenza diventa quindi:**

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

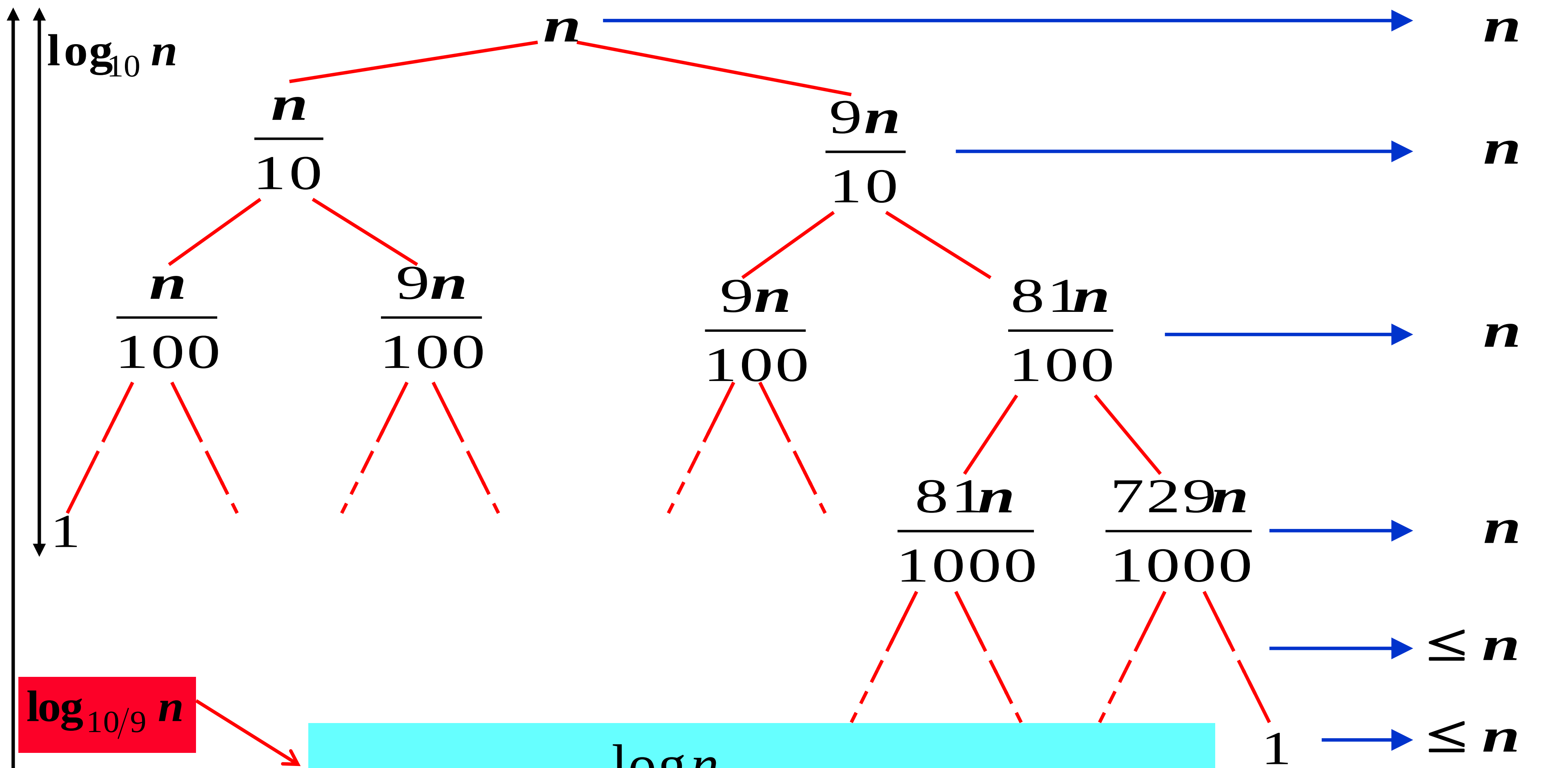
# Analisi di QuickSort: caso medio

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



# Analisi di QuickSort: caso medio

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




$$\log_{10/9} n = \frac{\log n}{\log 10/9} = c \log n = \Theta(\log n)$$

$\Theta(n \log n)$

## *Analisi di QuickSort: caso medio*

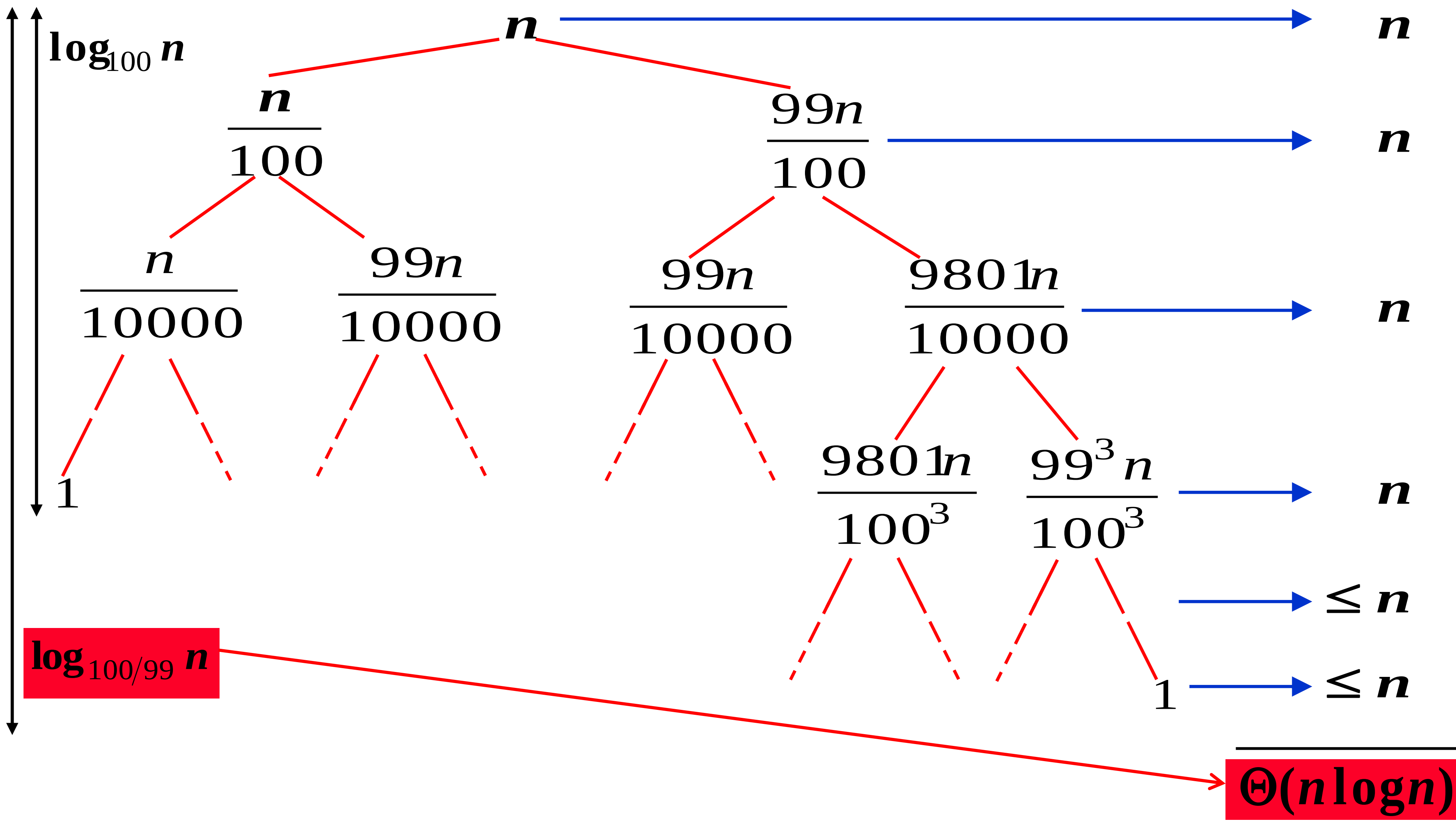
- **Supponiamo che ad ogni chiamata l'algoritmo `Partiziona` produca una partizione che è 99 volte l'altra (*partizionamento sbilanciato*)**

**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*

**Per fare un'analisi corretta del caso medio, è necessario definire una nozione chiara di *caso medio*.**

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

**In tal caso, se `QuickSort` è eseguito su un array di input casuale (*random*), ci aspettiamo che alcune *partizioni* siano *ben bilanciate* ed altre *mal bilanciate*.**

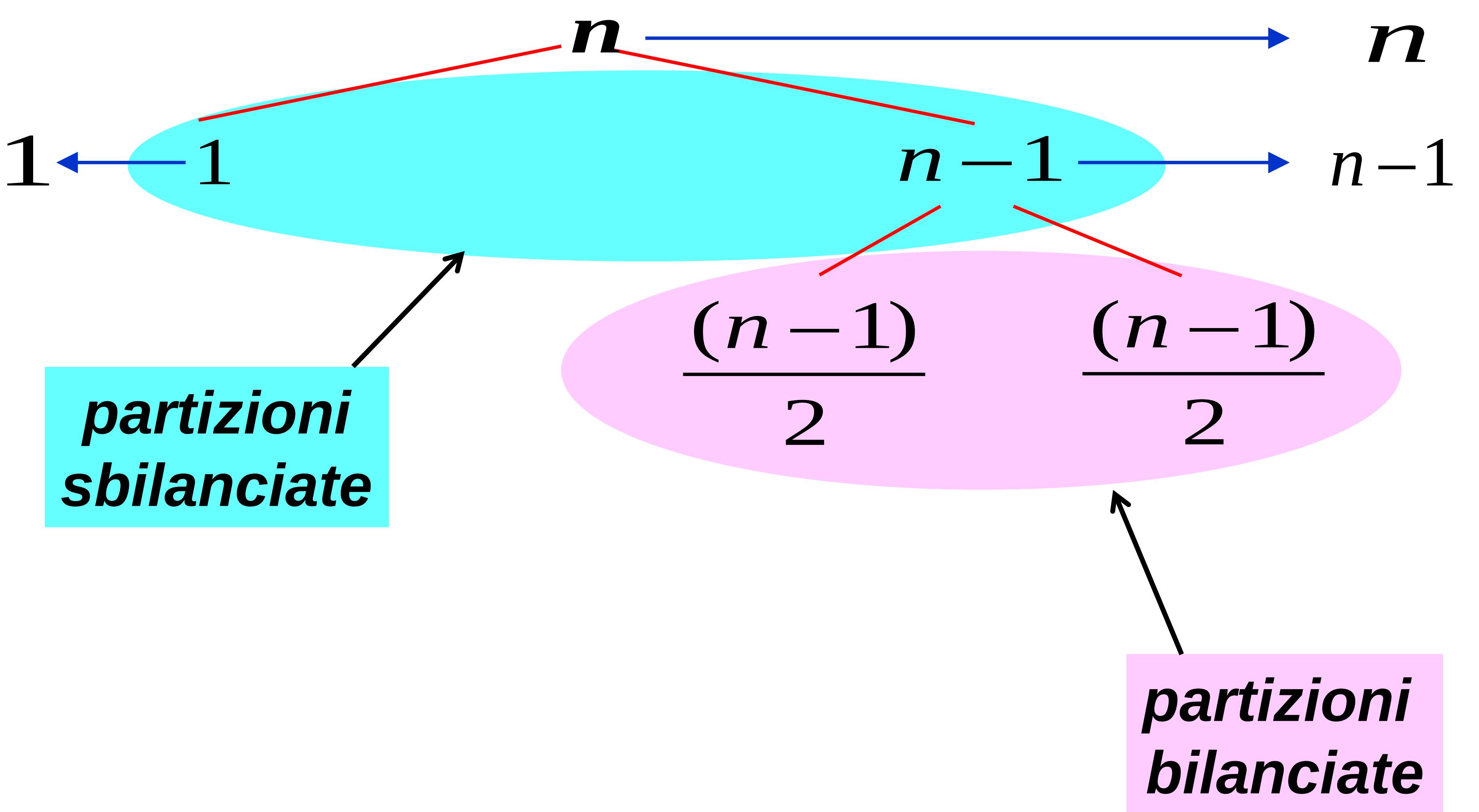
# *Analisi di QuickSort: caso medio*

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

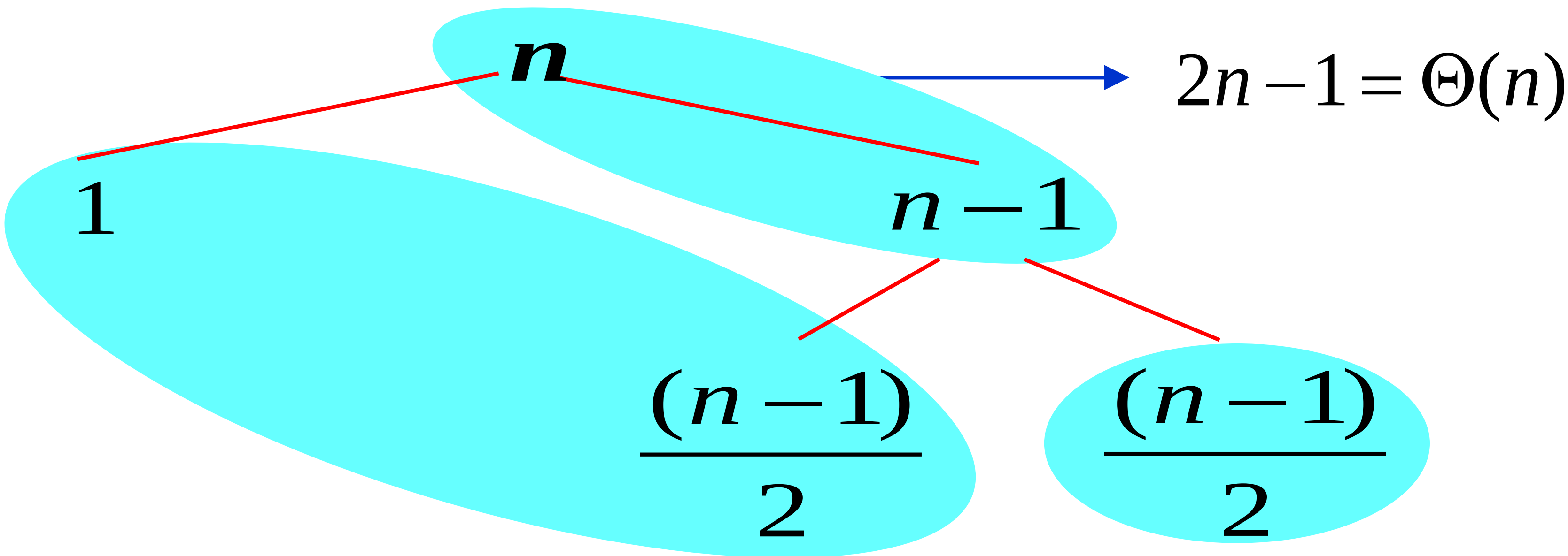
**Supponiamo che le *partizioni ben bilanciate* e quelle *mal bilanciate* si alternino nei diversi livelli dell'albero, cioè:**

- *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

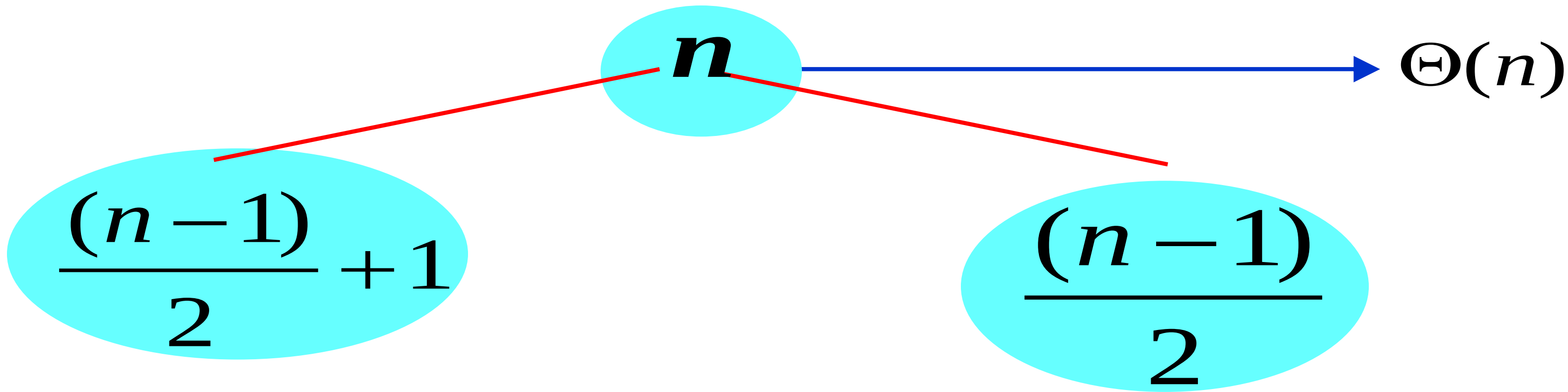


# Analisi di QuickSort: caso medio




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

# *Analisi di QuickSort: caso medio*




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

# Analisi di QuickSort: caso medio

***Dunque, supponendo 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, ora la notazione O-grande nasconde una costante maggiore che nel caso migliore***

# *Analisi di QuickSort*

***L'analisi che abbiamo fatto si basa sull'assunzione 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 importa!***

***ad esempio permutando in maniera casuale gli elementi dell'array in input***

# Analisi di QuickSort Random

```
int 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 valore intero casuale compreso tra *p* ed *r*.**

# *Analisi di QuickSort Random*

```
int 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

```
int 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 random di QuickSort presentata:***

- ***non modifica le prestazioni nel caso peggiore (che rimane quadratico) Perché?***
- ***né modifica le prestazioni nel caso migliore o medio.***
- ***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 (né del caso migliore).***

# *Analisi di QuickSort Random: Caso Peggiore*

`Partiziona` ***suddivide un array di dimensione $n$ in due partizioni di dimensioni (casuali) non note, 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 Peggiore

`Partiziona` *suddivide un array di dimensione $n$ in due partizioni di dimensioni (casuali) non note, 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:*

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

# Analisi di QuickSort Random: Caso Peggiore

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

**Usiamo il *metodo di sostituzione***

***Ipotizziamo $T(n) \le cn^2$***

**Sostituendo otteniamo**

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

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

# Analisi di QuickSort Random: Caso Peggiore

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

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

*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 Peggiore

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

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

La <u>derivata prima</u>:

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

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 <u>valore massimo</u> quando $q = 1$ o $q = n-1$.

# Analisi di QuickSort Random: Caso Peggiore

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

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

$$\le c\,( 1^2 + (n-1)^2 ) + \Theta(n)$$

$$\le c\,( n^2 - 2(n-1) ) + \Theta(n)$$

$$\le c\,n^2 - 2c\,(n-1) + \Theta(n)$$

# Analisi di QuickSort Random: Caso Peggiore

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

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

*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 (casuali) non note, che chiamiamo* $q$ *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:*

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

# 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

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

**Usiamo il *metodo di sostituzione***

***Ipotizziamo*** $T(n) \le c\, n \log n$

***Sostituendo otteniamo***

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

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

# Analisi di QuickSort Random: Caso Migliore

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

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

*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

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

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

La <u>derivata prima</u>:

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

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

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

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

$$\leq c\,(n \log n/2) + \Theta(n)$$

$$\leq c\,n \log n - c\,n + \Theta(n)$$

$$\leq c\,n \log n - c\,n + \Theta(n)$$

# Analisi di QuickSort Random: Caso Migliore

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

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

$$\le c\,(n \log n/2) + \Theta(n)$$

$$\le c\,n \log n - c\,n + \Theta(n)$$

$$\le c\,n \log n - c\,n + \Theta(n)$$

$$\le c\,n \log n$$

*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.***

- ***Per semplificare l'analisi, assumeremo che tutti gli elementi siano distinti.***

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

- ***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`?***

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

# *Analisi di QuickSort Random: Caso Medio*

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

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

- ***Essendo `A[p]` un elemento casuale dell'array, la probabilità che il rango di `A[p]` sia $i$ (con $i = 1,\ldots,n$) 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`***?***

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

- ***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***

***ciascun caso ha probabilità $1/n$***

***Nota: tutto questo è garantito solo se gli elementi sono tutti distinti!***

# *Analisi di QuickSort Random: Caso Medio*

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

- ***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 rispettivamente 1 ed $n$-1***

***con probabilità 1/$n$***

***Nota: tutto questo è garantito solo se gli elementi sono tutti distinti!***

# *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 $1/n$***

***Nota: tutto questo è garantito solo se gli elementi sono tutti distinti!***

# *Analisi di QuickSort Random: Caso Medio*

$$T(n) = \left( T(1) + T(n-1) + \quad \right) + \Theta(n)$$

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

***ciascun caso ha probabilità 1/n***

# *Analisi di QuickSort Random: Caso Medio*

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

# *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 Θ(n)!*** **(*Perché?*)**

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

# *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 <u>metodo di sostituzione</u>***

# 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***

$$\boxed{T(n) \leq a\, n \log n}$$

# Analisi di QuickSort Random: Caso Medio

***T(n) = O(n log n)***

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

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

# *Analisi di QuickSort Random: Caso Medio*

$$\boldsymbol{T(n) = O(n \log n)}$$

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

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

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

# *Analisi di QuickSort Random: Caso Medio*

***T(n) = O(n log n)***

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

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

***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

$$\boxed{T(n) = O(n \log n)}$$

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

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

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

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

# Analisi di QuickSort Random: Caso Medio

**T(n) = O(n log n)**

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

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

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

# Analisi di QuickSort Random: Caso Medio

**T(n) = O(n log n )**

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

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

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

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

# Analisi di QuickSort Random: Caso Medio

$\boldsymbol{T(n) = O(n \log n)}$

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

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

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

$$\leq \boldsymbol{an} \log \boldsymbol{n}$$

***Scegliendo a grande abbastanza da rendere a n/4 dominante su Θ(n)***

# Analisi di QuickSort Random: Caso Medio

**Possiamo concludere che**

$$\boxed{T(n) = O(n \log n)}$$

**A patto di dimostrare che**

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

# *Analisi di QuickSort Random: Caso Medio*

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

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

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

***Questo limite non è però sufficiente per risolvere la ricorrenza, ma quello che abbiamo calcolato sarà utile per trovane 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

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

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

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

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

# Analisi di QuickSort Random: Caso Medio

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

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

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

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

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

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

# *Analisi di QuickSort Random: Caso Medio*

***Possiamo concludere che:***

- ***nel <u>caso medio</u>, il tempo di esecuzione è:***

$$\boxed{T(n) = O(n \log n)}$$

- ***nel <u>caso migliore</u>, il tempo di esecuzione è:***

$$\boxed{T(n) = O(n \log n)}$$

- ***nel <u>caso peggiore</u>, il tempo di esecuzione è:***

$$\boxed{T(n) = O(n^2)}$$