

# Spanning Tree

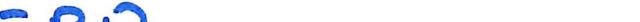
## Osservazione:

- BD, WP, TR

(H) (m)

$m = n^2$  di link  
 $n = n^2$  di entità

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


}


}

Strategia:

- al posto di usare l'intera rete G  
usiamo una sottorete  
per minimizzare la complessità  
di comunicazione

↓  
quale sottorete?

↓  
UN ALBERO

Attenzione ai costi per risolvere i problemi :

- costo di costruzione dell'albero
- costo originale del problema eseguito su di un albero

Esempio :

Il costo di BD sull'albero è esattamente  $n - 1$  messaggi

## Il problema dello Spanning Tree :

- costruire una sottorete t.c.
  - coinvolge TUTTE le entità
  - le entità sono connesse
  - è priva di cicli

La soluzione con un algoritmo distribuito  
richiede la conoscenza dell'albero  
all'interno della rete.

⇒ ogni entità vede una piccola parte  
dell'albero

NOTAZIONE :

- $\forall x \in E$  definiamo  
 $\text{Tree-}N(x) \subseteq N(x)$
- $\text{link}(\text{Tree-}N(x)) \ni (x, y)$   
 $\Updownarrow$   
 $y \in \text{Tree-}N(x)$
- $\text{Tree} = \bigcup_{x \in E} \text{link}(\text{Tree-}N(x))$

Risolviamo il problema Spanning Tree  
con le restrizioni RI

Il protocollo Shout:

- ogni entità vede solo i suoi figli  
tree- $N(x)$
- e tiene traccia del padre

La radice è data dall'entità che inizia  
il protocollo

Strategia di Shout : CHIEDI !

## Schemi:

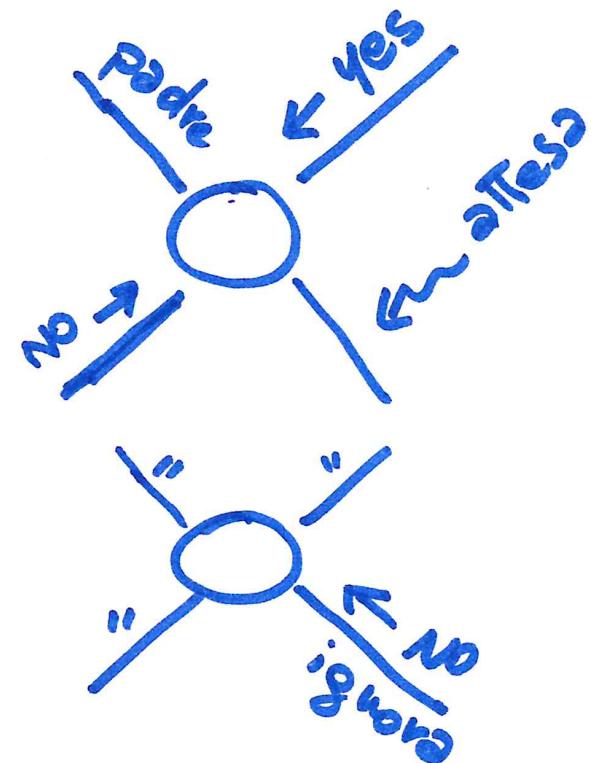
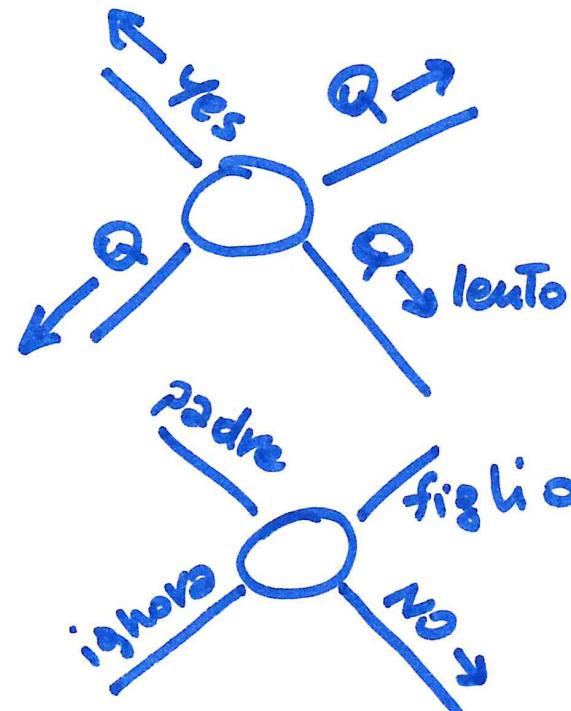
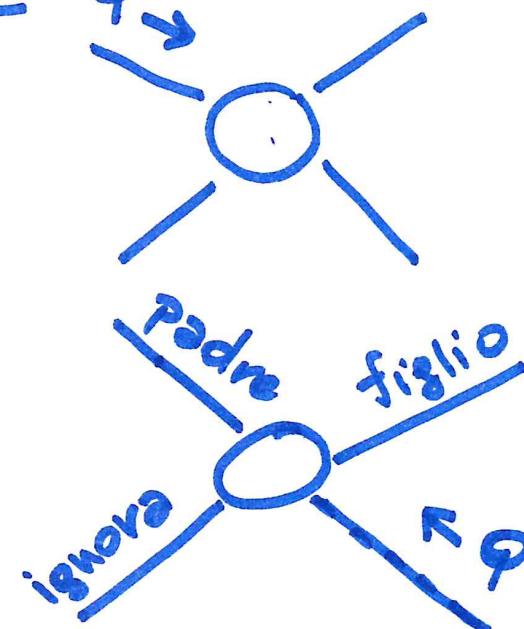
- Iniziatore s(root) spedisce Q ai suoi vicini e attende le risposte
- Ogni entità  $x \neq s$  che riceve Q per:
  - la prima volta risponde "Yes" e invia Q ai suoi vicini e si mette in attesa
  - una volta successiva alla prima risponde "no"
- Inoltre serve memorizzare l'entità padre e le entità che mi rispondono "yes"
- L'entità termina quando riceve tutte le risposte



Shout è sostanzialmente  
Flooding + Reply

Esempio :

Primo Q



Linee guida per definire il codice:

- mandare i messaggi: Q, Yes, No
- aggiornare variabili locali:  
root, parent, Tree-N( $\alpha$ ), counter
- aggiornare lo stato per raggiungere la terminazione

## Protocollo Shout

- Stati : Iniziatore, inattivi, attivi, finito

$S_{INIT} : \{ \text{Iniziatore}, \text{inattivi} \}$

$S_{FINAL} : \{ \text{Finito} \}$

- bisogna definire le azioni per  
iniz. , inat. , attivo

Iniziatore

Impulso Spontaneo

{ root = True

counter = 0

tree\_N(x) =  $\emptyset$

send (q) To N(x);

} become attivo;

Inattivo

Ricezione (q)

{ root = false;

parent = sender;

counter = 1;

tree- $N(x) = \{ \text{sender} \};$

send ("Yes") sender;

if counter =  $|N(x)|$  Then

become finite;

else

{ send(Q) To  $N(x) \setminus \text{sender};$

become active;

}

Attivo

Ricezione ( $\alpha$ )

{ send ("NO") to sender }

Ricezione ("yes")

{  $\text{Tree\_N}(\alpha) = \text{Tree\_N}(\alpha) \cup \{ \text{sender} \}$  ;

$\text{counter} = \text{counter} + 1$  ;

if ( $\text{counter} = |\text{N}(\alpha)|$ ) then  
become finito;

}

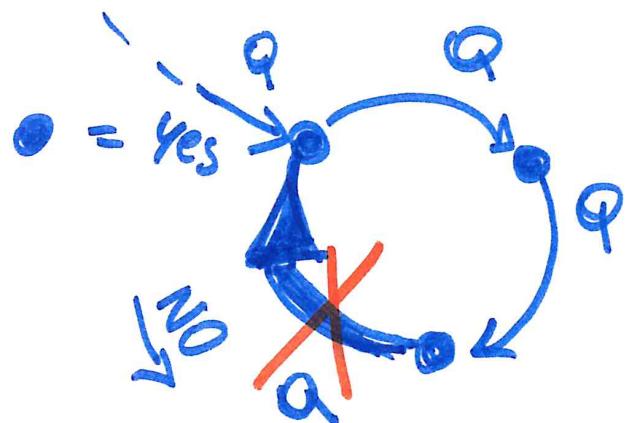
## Ricezione (No)

```
{   counter = counter + 1;  
    if (counter == IN(x)) then  
        become finito;  
}
```

## CORRETTEZZA DI SHOUT

- Termination: in assenza di errori riceve un n° di risposte pari ai Q inviati oliventandolo finito -

- Tutte le entità sono presenti: grazie al flooding di Q
- le entità sono connesse: grazie al fatto che al primo Q risponde "Yes"
- è priva di cicli: ogni entità risponde con un solo "Yes" tranne la radice che risponde sempre "No"



Costi :

$$M[\text{Shout}] = 2 M[\text{Flooding}]$$

$$= 2 [2m - (n-1)] = \sim 4m$$

$$\begin{aligned} T[\text{Shout}] &= T[\text{Flooding}] + 1 \\ &\leq d + 1 \end{aligned}$$

Lower Bound

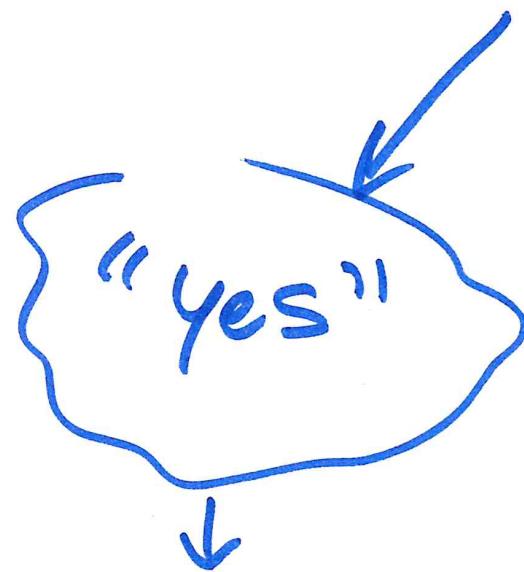
per SPT

$$M[\text{SPT / RI}] \geq m$$

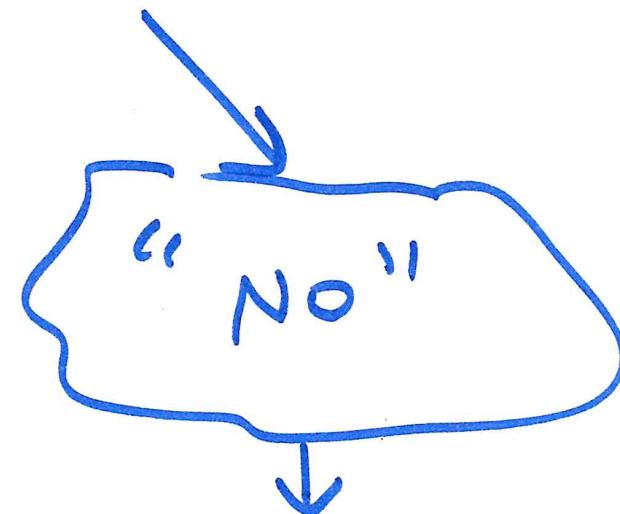
$$T[\text{SPT / RI}] \geq d$$

Shout migliorato : Shout+

- posso eliminare qualche msg?



si Tengono



Si possono  
eliminare

Infatti :

Se la risposta è un "NO" è perché  
l'entità ha già ricevuto in precedenza  
un Q a cui ha risposto "Yes" inviando  
contestualmente altri Q



Se ricevo un Q lo interpreto come  
un "NO"

Nuovo costo di comunicazione di Shout+

$$M[\text{Shout+}] = 2 \text{ m}$$

questo perché su ogni link  
viaggiano:

- o un Q seguito da Yes
- o un Q seguito da Q  
(o in contemporanea)

---

Altre soluzioni per lo Spanning Tree

- Si usa il protocollo TRAVERSAL

Tree = insieme degli archi =  
= insieme dei link su cui viaggiano  
i "Return"