Il problema ROUTING

Scopo: C'è una entità x che vuole spedire un msg all'entità y. Individuare un cammino in $\vec{G}$ da x a y.

Il problema Shortest Path

Scopo: Determinare il cammino migliore (di costo minimo) tra x e y in $\vec{G}$.

Applicazioni: COMUNICAZIONE, cruciale in una computazione di un sistema distribuito.

Osservazione: Per risolvere il primo problema è sufficiente che x faccia il BROADCAST del msg a Tutte le entità $y \in E$.

⇓

Soluzione inefficiente!

Quindi scegliamo un cammino Tra i Tanti possibili Tra x e y. Ovviamente se è il "migliore" possibile lo preferiamo

⇓

Secondo problema: richiede l'uso della memoria per registrare informazioni sui costi di $\vec{G}$ per ogni entità $x \in E$ al fine di calcolare i cammini minimi verso ogni altra entità!

Shortest Path

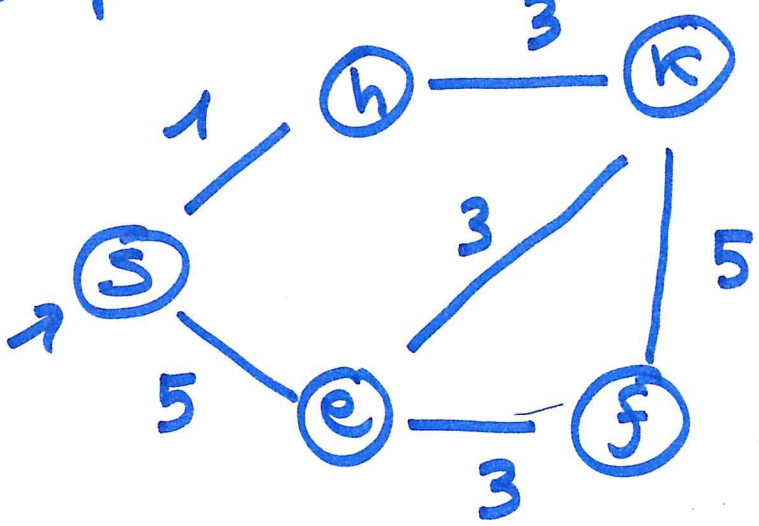
Risolviamo il problema sotto restrizioni : IR
(anche in questo caso $\forall x \in E$ abbiamo $id(x) =$ valore di x)

Le strategie per risolvere questo problema differiscono dal Tipo di info che le entità si tengono in memoria.

Definizione della FULL ROUTING TABLE
(ogni strategia alla fine ha bisogno di questa Tabella per risolvere il problema)

Esempio

G



FULL ROUTING TABLE per S

Destination	S. path	cost
h	(s, h)	1
k	(s,h) (h,k)	4
e	(s, e)	5
f	(s,e) (e, f)	8

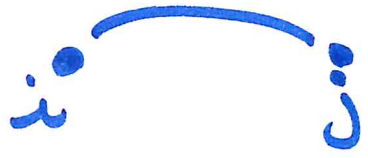
Protocollo GOSSIPING

Idea: ogni entità prima si costruisce la mappa del sistema G e all'occorrenza si calcola le righe della FULL ROUTING TABLE

Esempio:

MAP(G)

alla cella (i,j) abbiamo il peso dell'arco



G	s	h	k	e	f
s	0	1	–	5	–
h	1	0	3	–	–
k	–	3	0	3	5
e	5	–	3	0	3
f	–	–	5	3	0

Idea: per costruire MAP(G) in un ambiente distribuito è sufficiente che ogni entità x diffonde le proprie informazioni sui vicini ad ogni altra entità $y \in E$.

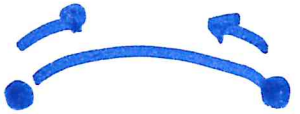
Map-Gossip:

- costruzione di un albero T per G
- ogni entità acquisisce dai vicini: id e i costi dei link
- ogni entità diffonde le sue informazioni a tutte le altre entità usando i link di T

Complessità di Map-gossip:

→ comunicazione = n^2 di msg

- Spanning Tree T: $O(m + n \log n)$ = il migliore
- acquis. info vicini: $2m$



- broadcast di info su T: $2m \cdot (n-1)$

$M[\text{Map Gossip}] \sim 2m \cdot n$

$\sim O(n^2)$ quando G è sparso

→ Tempo: difficile da calcolare

Protocollo Iterated-Construction

Strategia: ogni entità si costruisce la sua FULL ROUTING TABLE a più riprese senza usare MAP(G).

Inizialmente la F.R.T. contiene solo info dei vicini

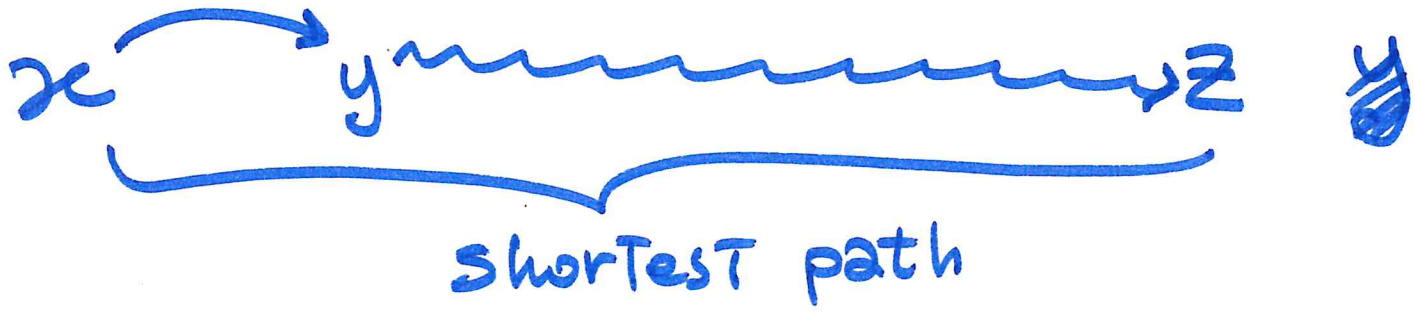
Esempio: F.R.T. iniziale di s:

Destination	S. Path	cost
h	(s,h)	1
k	—	∞ → costo infinito che verrà sostituito alle iterazioni succes.
e	(s,e)	5
f	—	∞

NOTA: La "F.R.T" di una entità x non necessariamente deve contenere l'intero shortest path per raggiungere una qualunque altra entità z. È sufficiente tenere traccia del vicino coinvolto nello shortest path per z.

$\forall z \in \mathcal{E}, z \neq x$

- il costo del s.p. per z
- il primo link dello s.p. che si traduce in un nodo nell'es: $y \in N(x)$



Def: Distance Vector.

È la F.R.T. ristretta alle colonne:
Destination e Cost e viene indicata con V.

Notazione: $V_x[z]$ = costo del cammino minimo da x a z.

Iterazione:
- ogni entità diffonde la propria "V" ai suoi vicini
- sulla base delle info che gli arrivano dai vicini stabilisce se ci sono stati trovati cammini minimi migliori di quelli che ha nella propria F.R.T. e in tal caso aggiorna la F.R.T.

Numero di iterazioni: $n-1$ Si può dim. x induz.

D: come fa una entità x ad individuare ad ogni iterazione il cammino minimo per raggiungere una certa entità z?

R: Sia $V_y^i[z]$ = costo del cammino minimo da y a z ottenuto da y alla i-esima iterazione.

$\Rightarrow$ alla i+1-esima iterazione questo costo arriva ai vicini di y e sia $y \in N(x)$.

Così x calcola alla i+1-esima iterazione:

$$w[z] = \text{Min}_{y \in N(x)} \{ \sigma(x,y) + V_y^i[z] \}$$

dove $\sigma(x,y)$ è il costo del link (x,y)

Se $w[z] < V_x^i[z] \Rightarrow x$ sceglie $w[z]$ come costo per il miglior s.p. per z aggiornando la sua F.R.T. e memorizza anche sotto la colonna shortest path il vicino $y \in N(x)$ che mi ha dato il costo $w[z]$.

VANTAGGI di questo approccio: la MEMORIA!
La F.R.T. richiede meno spazio di MAP(G)

Esempio di F.R.T. finale per l'entità S

Destination	s. path.	cost
h	h	1
k	h	4
e	e	5
f	e	8

Complessità di Iterated_Construction

→ n° messaggi

$$M[\text{It\_constr.}/IR] = \ell m \cdot n \cdot (n-1)$$

n ↓ # di msg per spedire V

→ Tempo

$$T[\text{It\_constr}/IR] = (n-1) \cdot \tau(n)$$

$\tau(n)$ ↓ Tempo ideale per trasm. V

$$\tau(n) = \begin{cases} O(1) & \text{costante } * \text{ quando } q \text{ consente msg lunghi} \\ O(n) & \end{cases}$$

CASO *: T è lineare in n e $M = O(m \times n)$