ALGORITMI DI INSTRADAMENTO (ROUTING)

Ogni nodo della rete ha un meccanismo in grado di decidere su quale sistema a code deve essere inviato il pkt in transito al fine di raggiungere efficientemente il destinatario

Il modo con cui questa decisione è presa si chiama politica di instradamento che è realizzata tramite un algoritmo di instradamento che può essere:

- centralizzato / distribuito
- statico / dinamico
-

- centralizzato: un'unica entità decide per tutti i nodi ⇒ questa deve conoscere lo stato dell'intera rete.
- distribuito: ogni nodo è partecipe e decide come instradare verso i vicini sulla base di info locali
- statico: ogni nodo ha tabelle in-out definite a priori
- dinamico: l'instradamento si adatta al traffico ⇒ si cerca di evitare la congestione di parte della rete.

L'algoritmo di instradamento può anche essere

- proattivo / reattivo
- uniforme / non uniforme

▵ proattivo : l'instradamento viene creato all'instaurarsi del collegamento

▵ reattivo : l'instradamento viene creato al momento del bisogno

▵ unif. / non unif : i nodi potrebbero avere ruoli diversi (es. WSN con cluster di nodi e cluster head) e la segnalazione per il routing dipende dal ruolo.

A seconda del tipo di instradamento c'è un diverso compromesso fra segnalazione e tempo di consegna dei pkt

$

gli algoritmi di instradamento risolvono problemi di ottimizzazione con funzioni di costo legate a: numero hop, costo fisico percorsi, ritardo complessivo o locale, throughput complessivo o locale. $C = \alpha_1 C_1 + \alpha_2 C_2 + \dots$

ALGORITMO DI DIJKSTRA

Risolve il problema della ricerca del percorso minimo (shortest path). Ne vediamo una versione centralizzata (1959).

$D(v)$ distanza / costo di v dalla sorgente s

$l(i,j)$ distanza / costo del percorso $i \to j$ (∞ se non esiste percorso)

S insieme di nodi selezionati ad una iterazione

i) inizializzazione

$S = \{s\}$, $\forall v \notin S$

$D(v) = l(s,v)$

ii) fra i nodi non già selezionati scelta di quello con distanza minore

$w \notin S \mid \forall z \notin S, z \neq w$

$D(w) \leq D(z)$

iii) selezione del nodo scelto

$S = S \cup \{w\}$

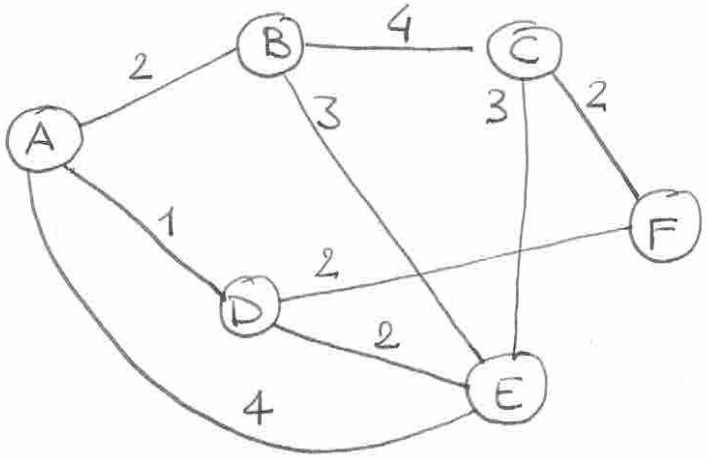
iv) calcolo nuove distanze considerato w

$\forall z \notin S \quad D(z) = \min\{D(z), D(w) + l(w,z)\}$

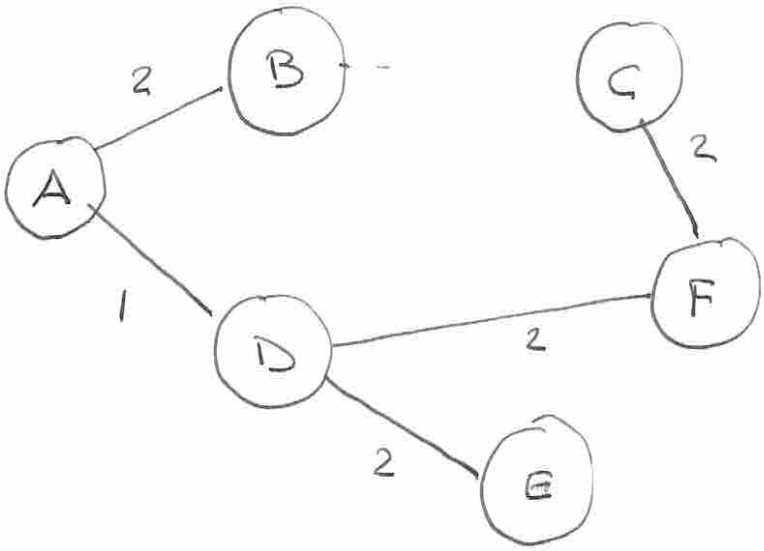
v) Se $|S| < M$ allora vai a (ii) altrimenti stop.

Esempio

Fisso sorgente $s = A$



N'	$D(B)$	$D(C)$	$D(D)$	$D(E)$	$D(F)$
$\{A\}$	2 (A-B)	∞	1 (A-D)	4 (A-E)	∞
$\{A,D\}$	2 $D(B)=2$ $D(D)+\ell(D,B)=\infty$ (A-B)	∞	1 (A-D)	3 $D(E)=4$ $D(D)+\ell(D,E)=3$ (A-D-E)	3 $D(F)=\infty$ $D(D)+\ell(D,F)=3$ (A-D-F)
$\{A,D,B\}$	2 (A-B)	6 (A-B-C)	1 (A-D)	3 (A-D-E)	3 (A-D-F)
$\{A,D,B,E\}$	2	6	1	3	3
$\{A,D,B,E,F\}$	2 (A-B)	5 $D(F)+\ell(F,C)=5$ (A-D-F-C)	1 (A-D)	3 (A-D-E)	3 (A-D-F)
$\{AD,B,E,F,C\}$	2 (A-B)	5 (A-D-F-C)	1 (A-D)	3 (A-D-E)	3 (A-D-F)



I percorsi sono a costo minimo, ma

- c'è un solo percorso sorg. - dest il che limita il throughput in caso di traffico sbilanciato

- difficoltà di adattamento a condizioni dinamiche del traffico

Nota: dall'algoritmo di Dijkstra deriva l'algoritmo di Viterbi per la decodifica dei codici convoluzionali.

Siamo in un contesto di rete di code e abbiamo definito una matrice di instradamento

$$\underline{\underline{I}} \quad (M+1) \times (M+1)$$

↑ sorg. (s) ↑ dest. (d)

i cui elementi I_{ij} rappresentano la prob. di andare da i a j.

$$\underline{\underline{I}} = \begin{bmatrix} I_{11} & I_{12} & \cdots & I_{1M} & I_{1d} \\ I_{21} & I_{22} & \cdots & I_{2M} & I_{2d} \\ & & \vdots & & \\ I_{M1} & I_{M2} & \cdots & I_{MM} & I_{Md} \\ I_{s1} & I_{s2} & \cdots & I_{sM} & I_{sd} \end{bmatrix}$$

$I_{sd} = \phi$ (si pone per la rete)

ALGORITMO DI BIFORCAZIONE (MULTIPATH ROUTING)

Si possono usare più linee contemporaneamente, se disponibili, e cercare di minimizzare il tempo medio W_s di permanenza di un pkt nella rete (di code).

$$W_s = \frac{1}{\lambda_s} \left[\sum_{i=1}^{M} \frac{f_i}{c_i - f_i} + \hat{\tau}_i \right] \simeq \frac{1}{\lambda_s} \sum_{i=1}^{M} \frac{f_i}{c_i - f_i}$$

trasc. $\hat{\tau}_i = \frac{\tau_i}{E\{F\}}$

$$c_i = \mu_i E\{F\}, \quad f_i = \lambda_i E\{F\}$$

$$\underline{c} = (c_1, c_2, \ldots, c_M) \qquad \underline{f} = (f_1, f_2, \ldots, f_M, f_s)$$

$W_S = W_S(\underline{\varsigma}, \underline{f}, \underline{\underline{I}})$

si deve risolvere un problema di ottimizzazione

$$\underline{\underline{I}}_{opt} = \arg\min_{\underline{\underline{\tilde{I}}}} W_S(\underline{\varsigma}, \underline{f}, \underline{\underline{\tilde{I}}})$$

s.t. vincoli

vincoli

- f_S tasso complessivo degli arrivi fissato
- rete non completamente connessa $\Rightarrow \underline{\underline{I}}$ ha alcuni elementi nulli a priori
- collegamenti con l'esterno (I_{si} e I_{id}) definiti a priori $\forall i$.
- velocità di trasm. dei collegamenti fissate ($\underline{\varsigma}$ noto)
- conservazione di flusso

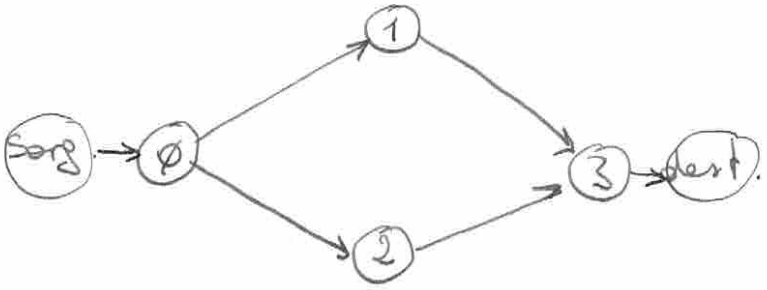
$$f_i = \sum_{j}^{M+1} I_{ji} f_j = I_{si} f_S + \sum_{j=1}^{M} I_{ji} f_j$$

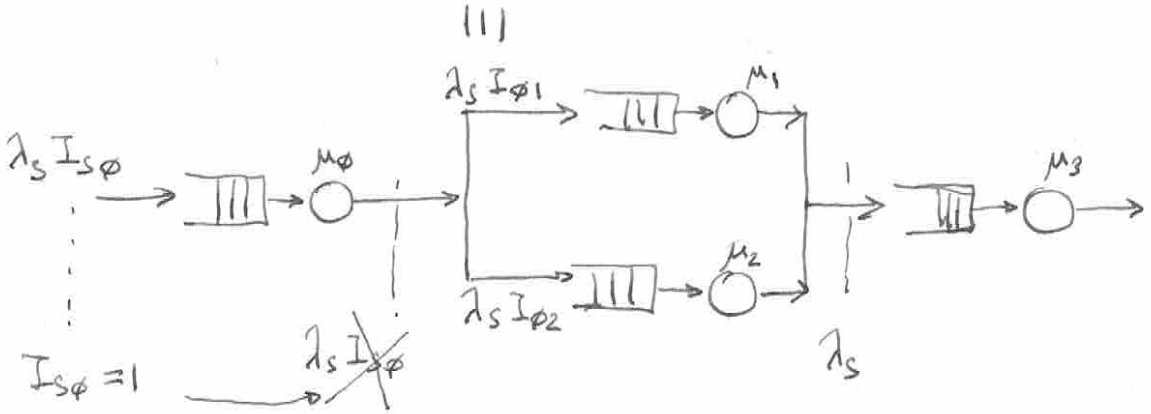
$$f_S = \sum_{j=1}^{M} I_{id} f_i$$

$$\Downarrow$$

$$\underline{f} = \underline{f}\,\underline{\underline{I}}$$

Esempio





$$I_{01} = \beta \quad , \quad I_{02} = 1-\beta \qquad \underline{\underline{I}} \quad 5 \times 5$$

$$\underline{\underline{I}} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ s \to \text{EXT} \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & \text{EXT}^{*d} \end{array} \\ \begin{bmatrix} 0 & \beta & 1-\beta & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$$\lambda_s W_s = \frac{f_s}{c_0 - f_s} + \frac{\beta f_s}{c_1 - \beta f_s} + \frac{(1-\beta) f_s}{c_1 - (1-\beta) f_s} + \frac{f_s}{c_3 - f_s} \qquad (\hat{c}_i \approx 0)$$

$$f_0 = f_s \quad , \quad f_1 + f_2 = f_0 (I_{01} + I_{02}) = f_s$$

$\hat{c}_i = c_i / f_s$ nota $\Rightarrow$

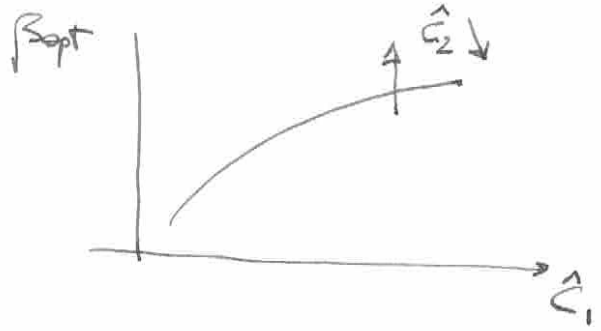
$$\boxed{\lambda_s W_s = \frac{1}{\hat{c}_0 - 1} + \frac{\beta}{\hat{c}_1 - \beta} + \frac{1-\beta}{\hat{c}_2 - 1 + \beta} + \frac{1}{\hat{c}_3 - 1}}$$

e si cerca β che la minimizza

$$\frac{d}{d\beta}\,\lambda_S W_S = \phi \iff \beta_{OPT} = \frac{\sqrt{\hat{c}_1 \hat{c}_2} + 1 - \hat{c}_2}{1 + \sqrt{\frac{\hat{c}_2}{\hat{c}_1}}} \in [\phi, 1]$$

Nota: la biforcazione ottima non dipende dalle caratteristiche dei nodi ϕ e 3 ma solo da quelle dei nodi fra cui biforca.

Dipende solo dal tasso di arrivo esterno (tramite f_S) e dalla velocità delle linee c_1 e c_2 su cui si fa biforcazione.



CUT THROUGH

Per aumentare le prestazioni si comincia a trasmettere un pkt in uscita prima di averlo ricevuto completamente ⇒ le informazioni sul destinatario devono essere nella parte iniziale del pkt.