Routing con distance vector

Algoritmo dinamico e distribuito. Converge ai percorsi minimi in maniera iterativa e asincrona.

Ciascun router:

1. calcola il distance vector $D_x = (d(x, x_1), \ldots, d(x, x_n))$ ($x_1, \ldots, x_n$ tutti i nodi della rete) e la tabella di routing;
2. invia il DV ai router vicini se è cambiato;
3. attende di ricevere un DV e torna a 1.

Ciascun router mantiene una tabella in cui in posizione x, y si trova il costo per raggiungere y da x; la riga x è il DV ricevuto dal router x. La tabella ha una riga per ogni vicino.

Inizialmente il DV di x contiene 0 associato a x, $c(x, y)$ per i vicini y e ∞ per gli altri. Per aggiornare il suo valore, si utilizza l'equazione di Bellman-Ford:

$$d(x, z) = \min_{y \in N(x)} \{c(x, y) + d(y, z)\}$$

Quindi ciascun nodo deve conoscere solo i costi di raggiungere i vicini e i loro DV.

Propagazione dei cambiamenti di costo

Se il costo di un link diminuisce, l'informazione viene diffusa rapidamente: i router ricalcolano i DV e li trasmettono a cascata.

Gli aumenti di costo invece possono propagarsi lentamente: se da

$$x \overset{4}{\longleftrightarrow} y \overset{1}{\longleftrightarrow} z$$

$c(x, y)$ aumenta a 60, y ricalcola $d(y, x) = 6$ perché sa che $d(z, x) = 5$; comunica il nuovo DV a z, che determina $d(z, x) = 7$, e così via finché non si arriva (lentamente) a $d(y, x) = 60$. Un caso peggiore è l'interruzione del collegamento tra x e y: $d(x, y) = \infty$ e si crea un ciclo tra y e z (*count to infinity*).

Soluzione: *split horizon with poisoned reverse*. Se z invia pacchetti a x attraverso y, il DV che z invia a y sarà alterato in modo da avere $d(z, x) = \infty$ — un percorso che passa da un router non viene pubblicizzato a quel router.

Costo e robustezza