Ruteo

Teoría de la Comunicaciones

20 de Septiembre de 2017

Forwarding vs Routing

Forwarding

Consiste en seleccionar un puerto de salida basándose en la dirección destino y en la tabla de ruteo.

Routing

Proceso por el cual se construye la tabla de ruteo.

Notación para los ejercicios

Network	Next hop
172.16.5.0/24	IF 0/1
10.4.2.0/27	IF 0/0
192.168.2.0/26	10.4.2.25
Default	10.4.2.25

Network (Red)	Next Hop (Próximo salto)
Red destino	- Interface de salida. (si la red esta directamente conectada a esa red) - Dirección IP del próximo salto. (Si la red destino es una red remota)

Routing: Construyendo la tabla de forwarding

Enrutamiento Estatico

- Genera carga administrativa y consume tiempo de administrador de red en redes grandes.
- El administrador debe configurar el enrutamiento en cada router de la red.
- El router no comparte su tabla de enrutamiento con los routers vecinos.
- Los routers no tienen capacidad de reacción ante un fallo/cambio en la red.

Enrutamiento Dinamico

- No genera mucha carga administrativa porque los routers aprenden a enrutarse de los demás routers de la red
- El router comparte su tabla de enrutamiento con los routers vecinos.
- Los routers tienen capacidad de reacción ante un fallo/cambio en la red

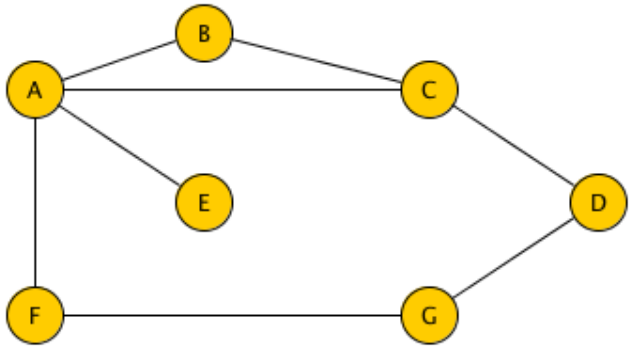
Los algoritmos de ruteo IGP (Internal Gateway Protocol)

NO ESCALAN

Distance Vector - RIP

1. Cada nodo construye un arreglo unidimensional conteniendo la *distancia* a todos los demás nodos. (1 par los directamente conectado, ∞ al resto)
2. Distribuye el arreglo a todos sus vecinos inmediatos.
3. Por cada mensaje que recibo, actualizo mi tabla de distancia sumando 1 a los nodos alcanzado por mi vecino. Si esa distancia es menor que la que yo conozco, la aprendo, sino lo descarto.

Distance Vector - Ejemplo



Nodo	Distancia al Nodo						
	A	B	C	D	E	F	G
A	0	1	1	∞	1	1	∞
B	1	0	1	∞	∞	∞	∞
C	1	1	0	1	∞	∞	∞
D	∞	∞	1	0	∞	∞	1
E	1	∞	∞	∞	0	∞	∞
F	1	∞	∞	∞	∞	0	1
G	∞	∞	∞	1	∞	1	0

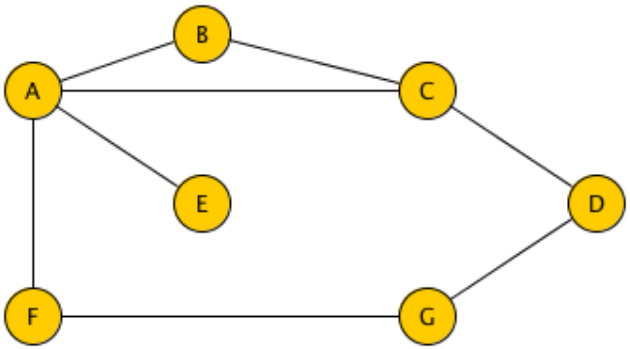
Vectores iniciales para cada nodo

¿Que cambios ocurririan si a **A** recibe el primer mensaje de **F** y luego de **C**?

Destino	**Costo**	**Proximo Salto**
B	1	B
C	1	C
D	∞	-
E	1	E
F	1	F
G	∞	-

Tabla parcial de ruteo inicial de **A**

Distance Vector - Ejemplo



Nodo	Distancia al Nodo						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

Vectores finales para cada nodo

Destino	**Costo**	**Proximo Salto**
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

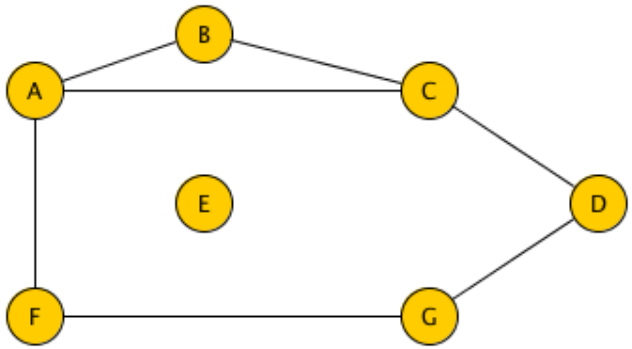
Tabla parcial de ruteo final de **A**

Distance Vector - Propiedades

- En ausencia de cambios en la topología, solo toma unos pocos intercambios de mensajes entre los vecinos antes de que cada nodo logre completar su tabla.
- Se dice que el proceso converge cuando todos los nodos obtienen una tabla de forwarding consistente.
- El algoritmo es distribuido, entonces ningún nodo tiene toda la información de la tabla, pero tiene una vision consistente de la red.
- Actualizaciones:
  - Periódicas, envían un update automáticamente en un intervalo de tiempo, aun si nada ha cambiado.
  - Disparados por cualquier mensaje que llegue que genere un cambio en la tabla, entonces la nueva tabla es transmitida.
  - Ante la caída de un nodo

Distance Vector - Cuenta al infinito

Se cae el enlace **A-E**



- Instante t_0 A(∞,E) , B(2,E) y C(2,E) y A informa su nuevo estado a B y C
- Instante t_1 A(∞,E) , B(∞,E) y C(2,E) y C se envia su estado a B
- instante t_2 B(∞,E) + C(2,E) = B(3,E) y se lo informa a A
- instante t_3 A(∞,E) + B(3,E) = A(4,E) y C(∞,E) (por el primer mensaje de A)
- instante t_4 A informa nuevo estado a C entonces, C(∞,E)+A(4,E)=C(5,E)

Distance Vector - Soluciones Parciales

Existen varias soluciones parciales al problema anterior:

Cota superior

Usar un número pequeño para aproximar infinito.

Hold down

Ante un fallo, los hosts ignoran las actualizaciones durante un periodo de tiempo hasta que se propague la noticia.

Split horizon

Cuando un nodo envía un mensaje de update a sus vecinos, no le envía las rutas que aprendió de ellos, de nuevo a ellos mismos.
Ej: Si B(2,E,A) entonces no envía a A esa ruta.

Split horizon with poison reverse

En este caso B le envia la ruta a A pero con información negativa B(∞,E)

Link State - OSPF

Se asume, igual que en distance-vector, que cada nodo conoce el estado y costo del enlace con todos sus vecinos.

Entonces cada nodo conoce como llegar a todos sus vecinos y si nos aseguramos que este conocimiento se **disemine** al resto de todos los nodos, entonces cada nodo tendrá la información necesaria acerca de la red para construir un mapa completo de la misma. Y esto es condición suficiente (no necesaria) para encontrar el **camino mínimo** a cualquier nodo en la red.

Link State - flooding

Es el proceso por el cual nos aseguramos que todos los nodos participantes del protocolo de ruteo consigan una copia del *link state* de todos los otros nodos.

Como sugiere la palabra flooding(inundación), la idea es enviar a todos mis vecinos mi información de *link state* y toda la información que reciba de mis vecinos **tambien** enviarla a todos los vecinos. Y este proceso continua hasta que la información haya llegado a todos los nodos.

Como nos aseguramos que la inundación sea confiable (todos reciban la copia mas reciente)?

Link State - LSP

Cada nodo crea un paquete LSP (link state package) que contiene los siquientes campos:

- ID del nodo que crea el paquete
- lista de todos los vecinos conectados a ese nodo y sus respectivos costos.
- número de secuencia
- tiempo de vida del paquete.

Los dos primero campos son justamente la información necesaria para poder armar el grado de la red.
Los últimos 2 son para poder realizar la inundación confiable

Link State - Reliable flooding

- El pasaje de LSP entre vecinos se asegura mediante mecanismos de ACK y retransmisión.
- Si recibo un LSP de X y **NO** lo tenia almacenado, lo almaceno y lo propago.
- Si recibo un LSP de X y **SI** lo tenia verifico el número de secuencia
  - si el recién llegado es mas nuevo que el que tenia almacenado, me lo quedo y ademas lo retransmito a todos **menos** al que me lo envío.
  - si el recién llegado es mas viejo o igual que el que tengo almacenado, lo descarto.
- El hecho de no volver a enviar el paquete de vuelta al que me lo envió,ayuda a que la inundación termine.
- Como los nodos envían la información a todos sus vecinos conectados, entonces la información mas reciente eventualmente alcanzara a todos los nodos.

Link State - LSP Cont.

- Al igual que RIP, los LSP se generan o bien periódicamente (en el orden de horas) o ante un cambio en la topología.
- El número de secuencia en los LSP no se supone que se agoten. (64 bits). En caso de reboot vuelven a 0, pero si encuentra un LSP suyo, actualiza su número de secuencia.
- El campo TTL de los LSP se decrementa cada vez que se recibe y se inunda al resto de los vecinos.
- EL campo TTL también sirve para añejar LPS almacenados en nodos. (cada un determinado tiempo se decrementa)
- los LSP que agotan un TTL se inundan para avisar que esa información es vieja y hay que descartarla.

Link State - Tabla de Forwarding

Terminada la inundación, se que todos los nodos de la red recibieron al menos 1 LSP de cada uno del resto de los nodos de la red. Esto significa que disponemos de la información suficiente para armar un grafo de la red completa.

Para armar la tabla de forwarding vamos a utilizar una variante del algoritmo de caminos mínimos de Djikstra llamada *forward search*.

Link State - Tabla de Forwarding

1. creo 2 listas **confirmados** y **tentativos**
2. Inicializo la lista **confirmados** con una entrada de mi mismo con costo 0,
3. llamaremos **proximo** al nodo recién agregado a la lista **confirmados**
4. por cada **vecino** de **proximo**, calculo el **costo** de alcanzar a **vecino** como la suma de los costos de alcanzar a **proximo** mas el costo de **proximo** a **vecino**.
    1. Si **vecino**.no esta ni en **confirmados** ni **tentativos**, agrego a **tentativos** (**vecino**,**costo**,NextHop) donde NextHop es la direccion usada para alcanzar **proximo**.
    2. Si **vecino**. esta en **tentativos**, y el costo es menor, lo reemplazo.
5. si hay elementos en **tentativos** tomo el de menor costo y lo muevo a **confirmados** y vuelvo al punto 2. Si esta vacío TERMINO.

Link State - Ejemplo forward search

1. creo 2 listas **confirmados** y **tentativos**
2. Inicializo la lista **confirmados** con una entrada de mi mismo con costo 0.
3. llamaremos **proximo** al nodo recién agregado a la lista **confirmados**

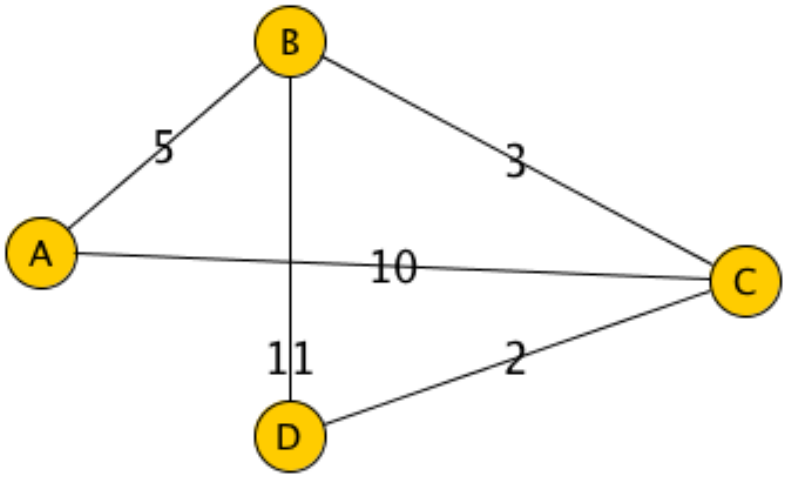


Tabla en Nodo **D**

Paso	**confirmado**	**tentativo**
1	(D,0,-)	

Como D es el único miembro en confirmado, miro sus LSPs

Link State - Ejemplo forward search

4 por cada **vecino** de **proximo**, calculo el **costo** de alcanzar a **vecino** como la suma de los costos de alcanzar a **proximo** mas el costo de **proximo** a **vecino**.

1. Si **vecino**.no esta ni en **confirmados** ni **tentativos**, agrego a **tentativos** (**vecino**,**costo**,NextHop) donde NextHop es la direccion usada para alcanzar **proximo**.
2. Si **vecino**. esta en **tentativos**, y el costo es menor, lo reemplazo.

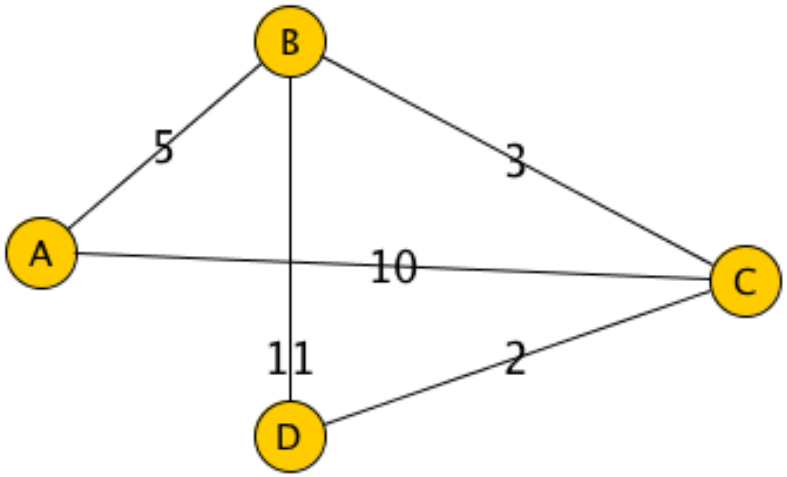


Tabla en Nodo **D**

Paso	**confirmado**	**tentativo**
1	(D,0,-)	
2	(D,0,-)	(B,11,B) (C,2,C)

Los LSP de D dicen que puede alcanzar B atravez de B con costo 11, que es mejor que nada entonces lo agrega a tentativo, igual que C

Link State - Ejemplo forward search

5 si hay elementos en **tentativos** tomo el de menor costo y lo muevo a **confirmados** y vuelvo al punto 2. Si esta vacío TERMINO.

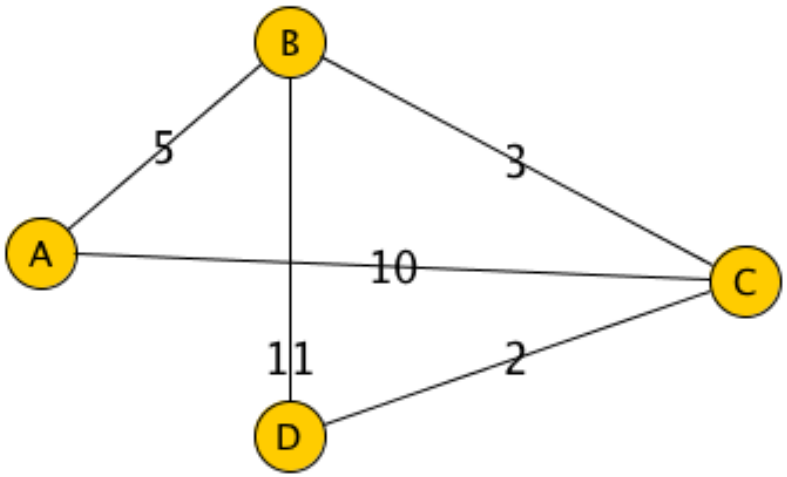


Tabla en Nodo **D**

Paso	**confirmado**	**tentativo**
1	(D,0,-)	
2	(D,0,-)	(B,11,B) (C,2,C)
3	(D,0,-) (C,2,C)	(B,11,B)

Link State - Ejemplo forward search

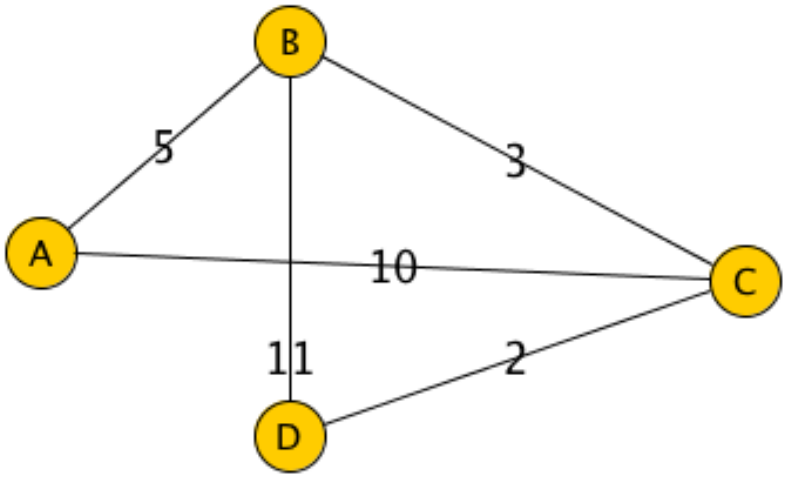


Tabla en Nodo **D**

Paso	**confirmado**	**tentativo**
1	(D,0,-)	
2	(D,0,-)	(B,11,B) (C,2,C)
3	(D,0,-) (C,2,C)	(B,11,B)
4	(D,0,-) (C,2,C)	(B,5,C) (A,12,C)
5	(D,0,-) (C,2,C) (B,5,C)	(A,12,C)
6	(D,0,-) (C,2,C) (B,5,C)	(A,10,C)
7	(D,0,-) (C,2,C) (B,5,C) (A,10,C)	

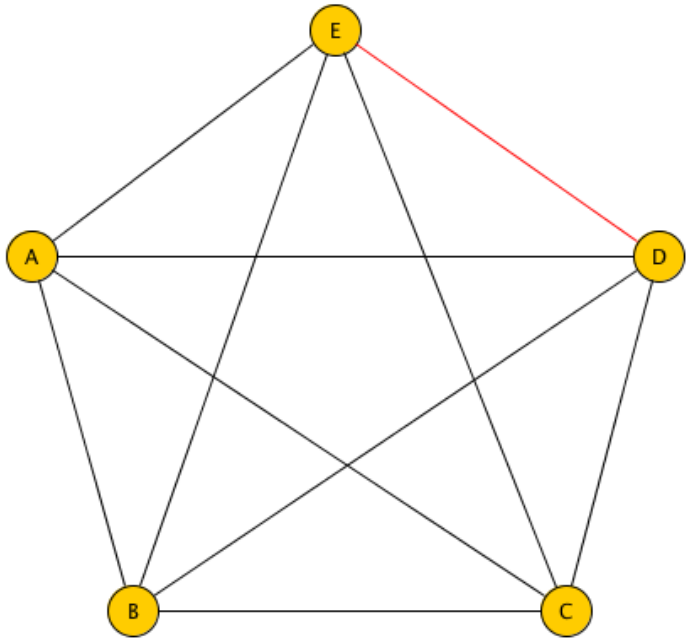
Resumen

Distance-Vector Envia Informacion de **TODA** la red **SOLO** a sus vecinos directos

Link-State Envia Informacion **SOLO** de sus vecinos directos a **TODA** la red

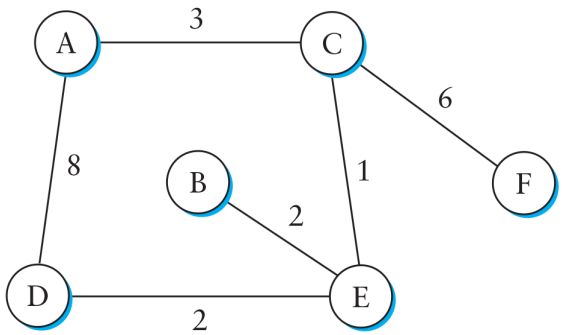
Para discutir en clase

En cada red, cual es el trafico generado por RIP y OSPF en el segmento D-E?



Ejercicio 1

En la red de la figura los enlaces están etiquetados con los costos relativos.



a. Mostrar la tabla de forwarding para cada nodo. Cada tabla en cada nodo debe reflejar la ruta de menor costo para el envío de un paquete a un determinado destino.

b. ¿De qué maneras se pueden llenar esas tablas?

c. En el caso de ruteo dinámico, ¿Qué problemas se resuelven, además del llenado de las tablas?

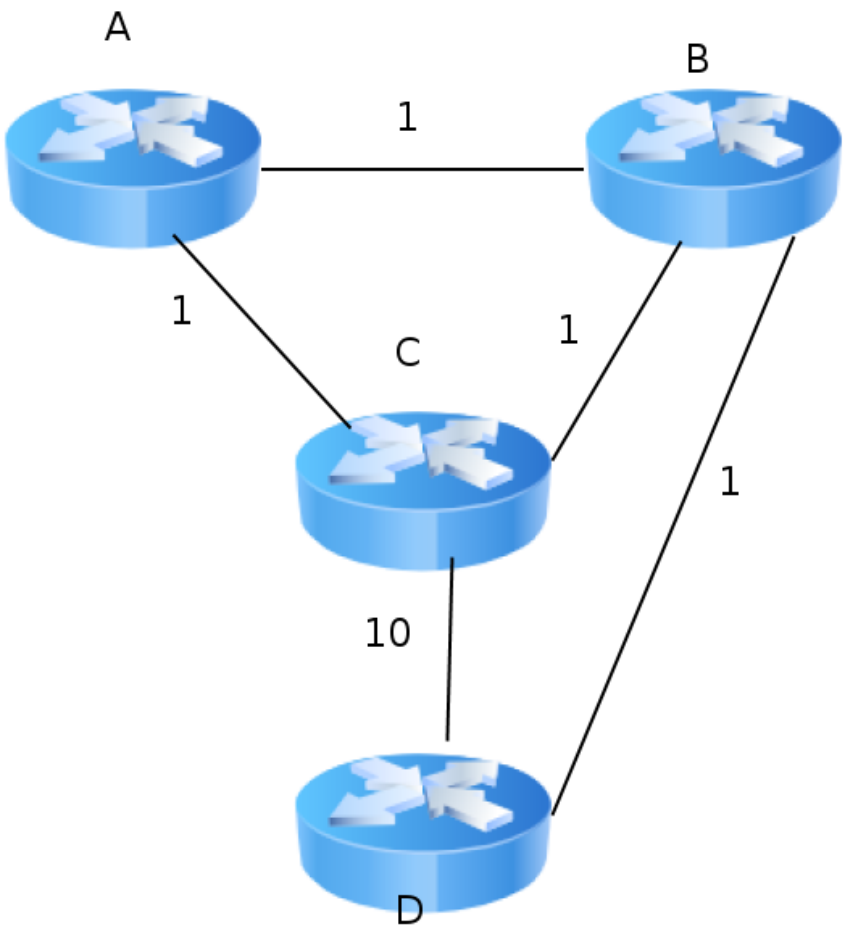
Ejercicio 4

Para un protocolo de vector de distancias corriendo en la red de la figura:

a. ¿Cuántas corridas de intercambio de mensajes se necesitarían para llegar a un estado de convergencia?

b. Suponga la caída de D. ¿Qué diversos posibles escenarios se le ocurren de intercambio de mensajes respecto a la convergencia?

c. Indicar y explicar tres formas de prevenir, anular y/o disminuir las anomalías respecto al retardo de convergencia.



Ejercicio 10 - Parcial

Dada la red de la figura, suponer que el protocolo de ruteo utilizado es OSPF. Se pide:

a. Mostrar todos los mensajes (el contenido de los campos relevantes) que recibe A hasta que la red converge.

b. Explicar cómo A construye su tabla de ruteo a partir de los mensajes recibidos.