

Enrutamiento (1)

Area de Ingeniería Telemática
<http://www.tlm.unavarra.es>

Arquitectura de Redes, Sistemas y Servicios
3º Ingeniería de Telecomunicación

Basadas en el material docente de Lawrie Brown sobre el libro de
William Stallings (Data and Computer Communications)

Temario

- Introducción
- Arquitecturas, protocolos y estándares
- Conmutación de paquetes
- Conmutación de circuitos
- Tecnologías
- Control de acceso al medio en redes de área local
- Servicios de Internet

Temario

1. Introducción
2. Arquitecturas, protocolos y estándares
3. **Conmutación de paquetes**
 - Principios
 - **Problemas básicos**
 - Como funcionan los routers (Nivel de red)
 - **Encaminamiento (Nivel de red)**
 - Transporte fiable (Nivel de transporte en TCP/IP)
 - Control de flujo (Nivel de transporte en TCP/IP)
 - Control de congestión (Nivel de transporte en TCP/IP)
4. Conmutación de circuitos
5. Tecnologías
6. Control de acceso al medio en redes de área local
7. Servicios de Internet

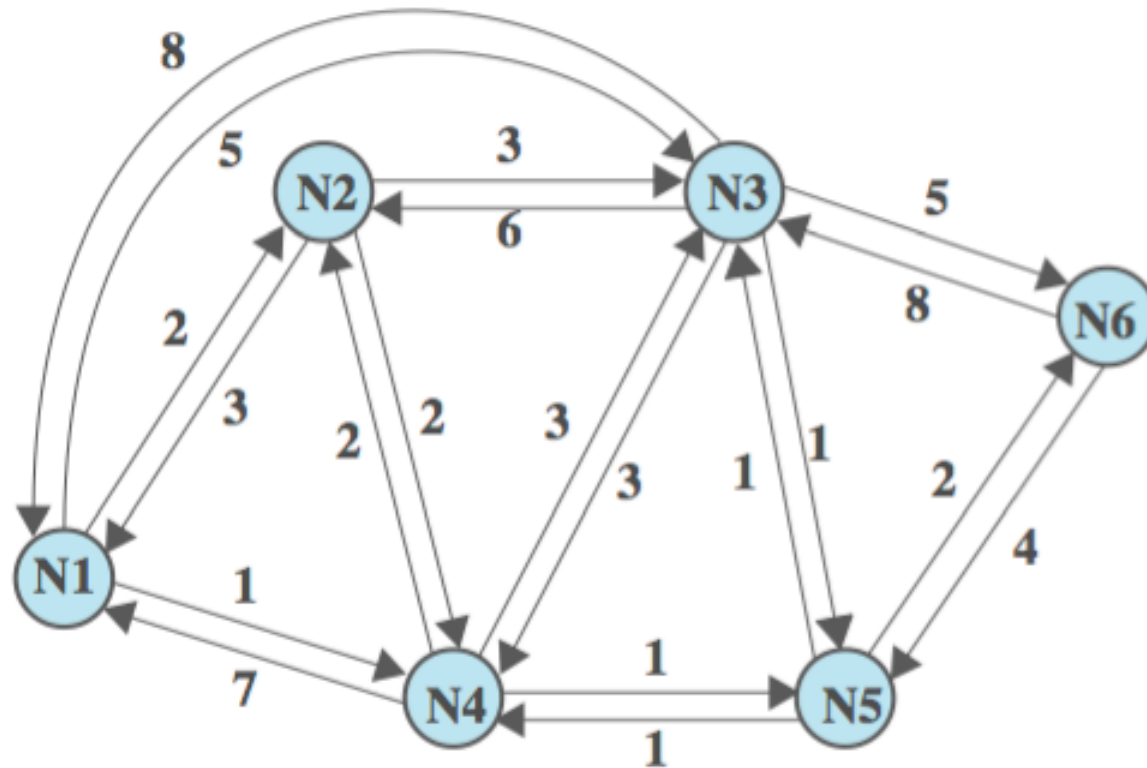
Enrutamiento en redes de paquetes

- Problema clave
 - Elegir camino a través de la red de nodos
 - Características
 - corrección
 - simplicidad
 - robustez
 - estabilidad
 - justicia
 - optimalidad
 - eficiencia
- compromiso
- compromiso
-

Criterio de prestaciones

- Elegido para comparar rutas
- El más sencillo “minimum hop count” = mínimo número de saltos
 - Encontrar la ruta de con menor numero de saltos(nodos)
- Se puede generalizar como mínimo coste (least cost)
 - Encontrar la ruta con menor coste (peso del camino)
 - Si el peso de cada enlace es 1 es equivalente al anterior

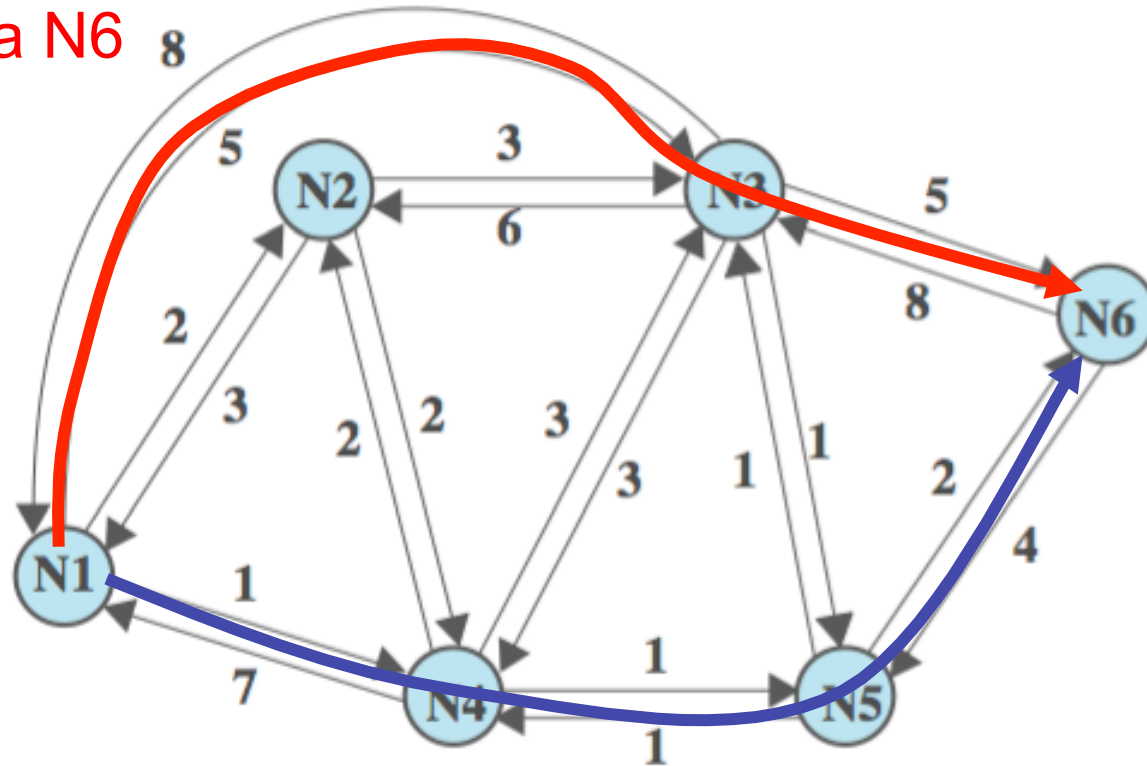
Ejemplo



En general grafo dirigido
con enlaces con pesos

Ejemplo

Mínimo número de saltos (hops)
De N1 a N6



Mínimo coste
De N1 a N6

Tiempo y lugar de decisión

- Tiempo
 - Por paquete o por circuito virtual
 - Fija por destino o cambia según el estado de la red
- Lugar
 - distribuida - cada nodo decide
 - centralizada
 - origen (source routing) el nodo que origina la información elige todo el camino

Información de la red

- Normalmente el enrutamiento requiere conocer información sobre la red (no siempre)
 - Enrutamiento distribuido
 - Conocimiento local, información de nodos vecinos...
 - Enrutamiento centralizado
 - Información de todos los nodos
- Actualización de la información sobre la red
 - ¿Cuándo?
 - Enrutamiento estático - nunca se actualiza
 - Enrutamiento adaptativo - actualizaciones regulares

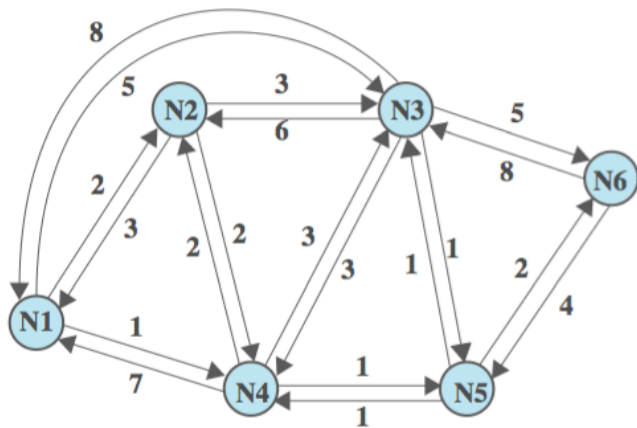
Estrategias de enrutamiento

- Enrutamiento estático
- Enrutamiento por inundación
- Enrutamiento aleatorio
- Enrutamiento adaptativo

Enrutamiento estático

- Ruta única y permanente para cada destino
- Se calcula con algún algoritmo de mínimo coste
- La ruta es fija (la configura el administrador de la red)
 - Al menos hasta que haya un cambio de topología que habrá que configurar nuevas
 - No puede responder a los cambios en el tráfico
- Ventaja: simplicidad
- Desventaja: falta de flexibilidad

Ejemplo con enrutamiento estático



CENTRAL ROUTING DIRECTORY

		From Node					
		1	2	3	4	5	6
To Node	1	—	1	5	2	4	5
	2	2	—	5	2	4	5
	3	4	3	—	5	3	5
	4	4	4	5	—	4	5
	5	4	4	5	5	—	5
	6	4	4	5	5	6	—

Node 1 Directory

Destination	Next Node
2	2
3	4
4	4
5	4
6	4

Node 2 Directory

Destination	Next Node
1	1
3	3
4	4
5	4
6	4

Node 3 Directory

Destination	Next Node
1	5
2	5
4	5
5	5
6	5

Node 4 Directory

Destination	Next Node
1	2
2	2
3	5
5	5
6	5

Node 5 Directory

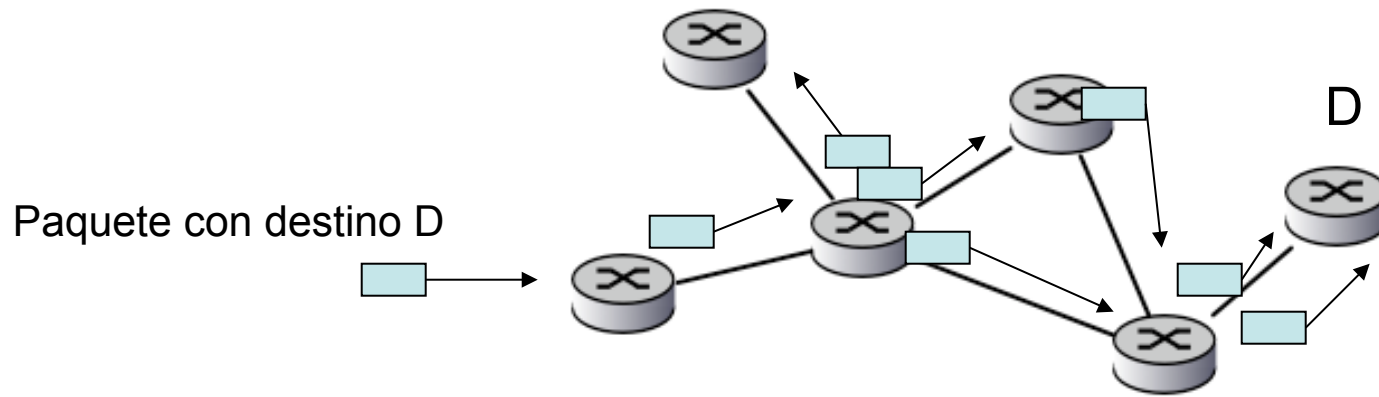
Destination	Next Node
1	4
2	4
3	3
4	4
6	6

Node 6 Directory

Destination	Next Node
1	5
2	5
3	5
4	5
5	5

Enrutamiento por Inundación

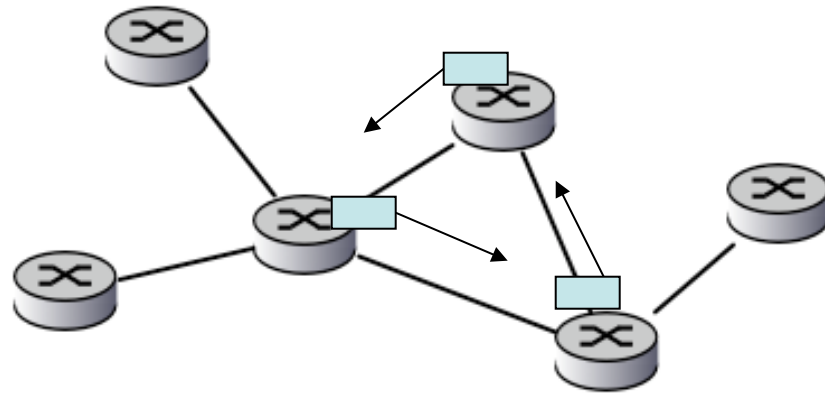
- Si un nodo recibe un paquete lo envía a todos sus vecinos (menos a aquel que se lo ha enviado)
- Simple, pero funciona
- Eventualmente múltiples copias llegarán al destino
- No requiere información de la red para funcionar
- Necesitamos identificar cada paquete para distinguir si un paquete lo hemos recibido ya o no. (Pero es facil, basta con poner un número de secuencia en el paquete)
- Algún problema más?



Enrutamiento por Inundación

Problemas:

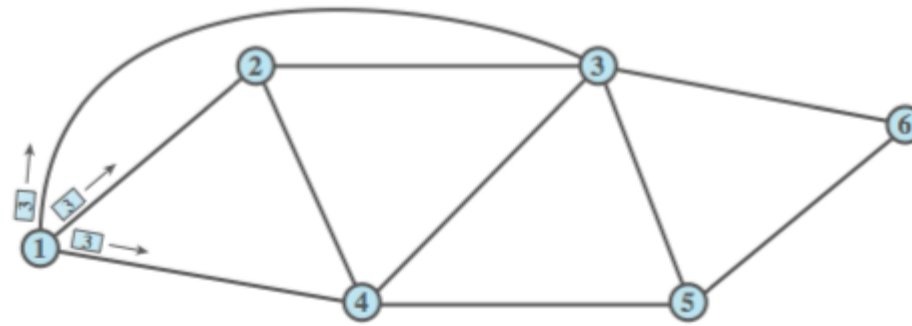
- Los ciclos crean tráfico infinito



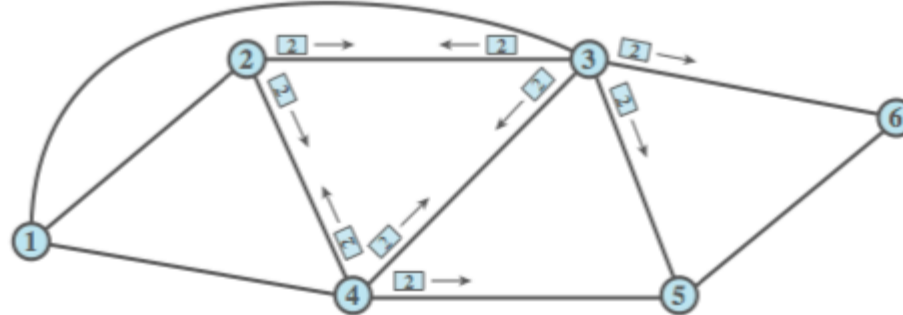
¿Cómo limitamos los el tráfico en los ciclos?

- Los nodos podrían recordar los paquetes que han reenviado y no volver a reenviar de nuevo (Cuanto tiempo deben recordarlos? Que problema hay si lo recuerdan mucho tiempo?)
- Se puede incluir un numero máximo de saltos en cada paquete e ir decrementando en cada salto (recuerde TTL de IP)

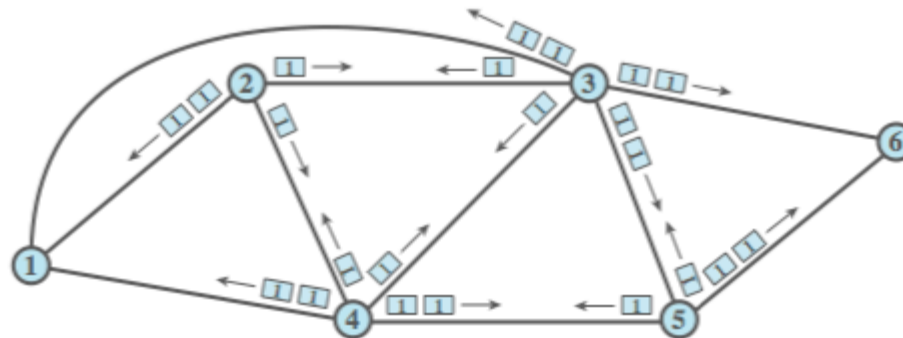
Ejemplo



(a) First hop



(b) Second hop



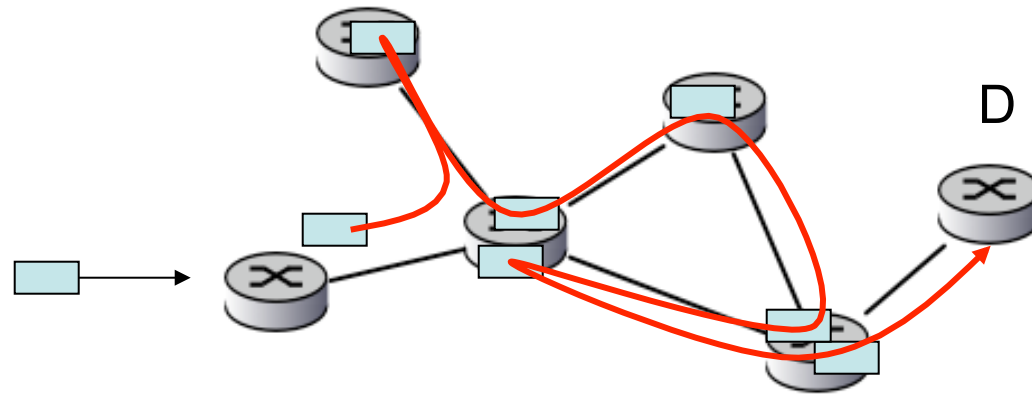
(c) Third hop

Propiedades de la inundación

- Todos los posibles caminos se prueban
 - Muy robusto
- Al menos un paquete viaja por el camino más rápido
 - Muy útil para establecer circuitos virtuales
- Todos los nodos son visitados
 - Útil para distribuir información a múltiples destinos (Broadcast y Multicast)
- Desventaja: mucho tráfico generado (incluso con limitaciones)

Enrutamiento aleatorio

- La simplicidad de la inundación con mucha menos carga
- Cada nodo que debe reenviar un paquete:
 - Elige uno de los enlaces de salida y lo envía por ese
 - La selección puede ser al azar o bien ir eligiendo uno cada vez (Round Robin)
 - Una refinamiento es asignar una probabilidad diferente de ser elegido a cada enlace
- No requiere información de la red



Enrutamiento aleatorio

- Al final acaba llegando al destino
- Aunque la ruta aleatoria normalmente no es ni la de menos salto ni la de menos coste
- Si los paquetes tienen un número de saltos limitado el enrutamiento aleatorio tiene una probabilidad de entregar el paquete menor que 1
- Puede parecer malo pero hay ocasiones en las que es útil
 - Ventajas: muy simple y poca carga (comparado con la inundación) y visita un número grande de nodos (aunque menos que la inundación)
 - Desventajas: no siempre llega, normalmente no llega por el camino más corto
- En qué situación es útil esto?

Enrutamiento adaptativo

- Usado por prácticamente todas las redes de conmutación de paquetes
- Las decisiones de enrutamiento cambian conforme cambia el estado de la red, debido a fallos y desconexiones de enlaces o a la congestión
- Necesita información de la red
- Desventajas
 - Decisiones más complejas
 - Compromiso entre información de la red utilizada y trafico extra introducido por el enrutamiento (mejor información más capacidad de red desperdiciada en trafico de enrutamiento)
 - Compromisos de estabilidad
 - Reaccionar muy rápido puede causar oscilaciones y desorden o ciclos momentaneos
 - Reaccionar muy lento = información desactualizada, perdidas por enlaces caidos

Enrutamiento adaptativo

Ventajas

- Mejores prestaciones
- Puede ayudar al control de congestión

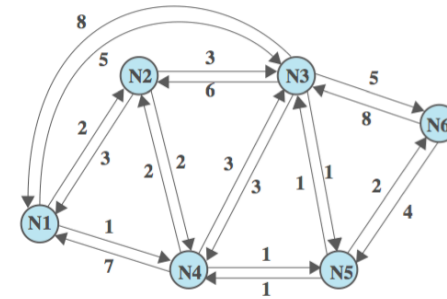
- Pero es un sistema complejo por lo que es difícil conseguir las ventajas teóricas
 - La mayoría de las redes de paquetes han sufrido problemas de enrutamiento debido a fallos en los sistemas de enrutamiento adaptativos y han cambiado de sistemas de enrutamiento a lo largo del tiempo

Clasificación del enrutamiento adaptativo

- Según la fuente de información
 - Local (aislado)
 - Enviar por el enlace mas descargado (menos paquetes en cola)
 - Puede incluir bias de enrutamiento estatico
 - Se usa muy poco ya
 - Nodos vecinos
 - Información sobre retardo o disponibilidad (caidas de enlaces) se envían a los vecinos
 - Todos los nodos
 - La información sobre los enlaces y su retardo se envía a todos los demas nodos (distribuido) o a un nodo central que decide para todos (centralizado)

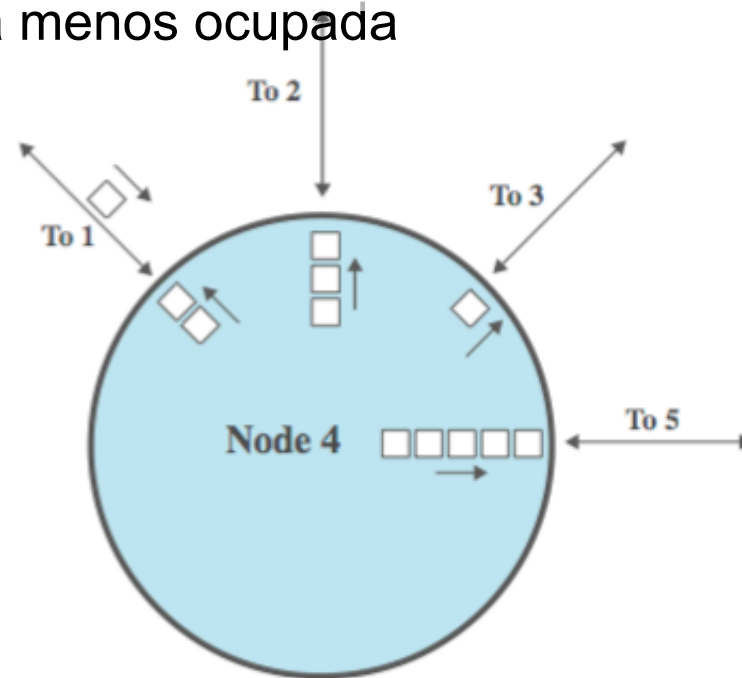
Ejemplo

- Encaminamiento aislado
 - Envío el paquete por la cola menos ocupada



Node 4's Bias
 Table for
 Destination 6

Next Node	Bias
1	9
2	6
3	3
5	0



- Añadir un bias
 - Ejemplo: para ir al destino 6 añadimos los bias de la tabla
 - Sabemos que al destino 6 el camino más corto es por 5 (por enrutamiento estático)

Algoritmos de coste mínimo

- Básicos en los algoritmos de enrutamiento
Permiten encontrar caminos en un grafo con enlaces con pesos
 - Minimizan el numero de saltos si los enlaces tienen peso 1
 - Minimizan el “coste” del camino si etiquetamos a los caminos con un peso. (inversamente proporcional a la capacidad, otras métricas con el retardo)
- El coste del camino entre dos nodos es la suma de todos los enlaces atravesados
 - Normalmente enlaces bidireccionales
 - Con un coste en cada dirección
- Para cada par de nodos encontrar el camino entre ellos con el menor coste
- Algoritmos básicos:
 - **Dijkstra**
 - **Bellman-Ford**

Algoritmo de Dijkstra

- Encontrar caminos más cortos desde un nodo de origen dado a todos los demás nodos
- Construyendo los caminos en orden creciente de longitud de camino
- El algoritmo funciona en iteraciones
 - En cada iteración se añade un nodo con el siguiente camino más corto
 - En la iteración k conocemos los caminos a los k nodos más cercanos (conjunto T)
- El algoritmo termina cuando todos los nodos están en el conjunto T

Algoritmo de Dijkstra

- Variables del algoritmo
- Conjunto de nodos $E = \{n_i \text{ nodos en el grafo}\}$
- Pesos de los enlaces

$w(n_i, n_j)$ peso del enlace de n_i a n_j

en general distinto de $w(n_j, n_i)$

$w(n_i, n_j) = \text{infinito}$ si no hay enlace

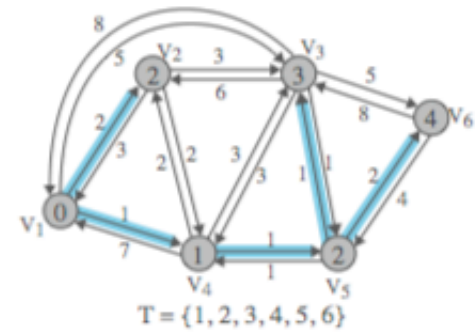
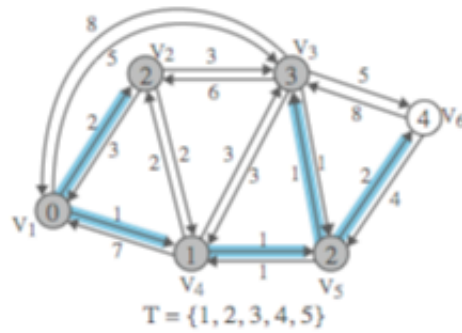
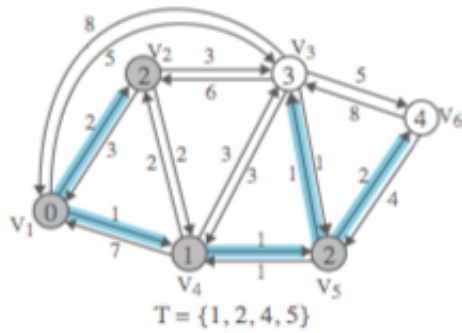
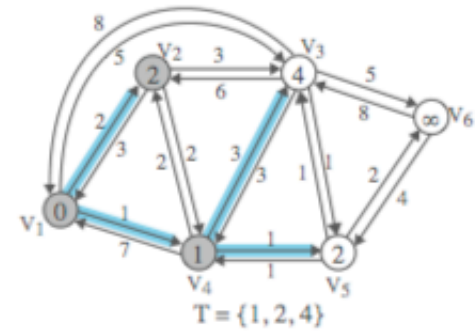
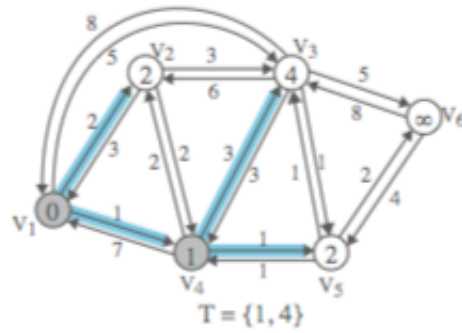
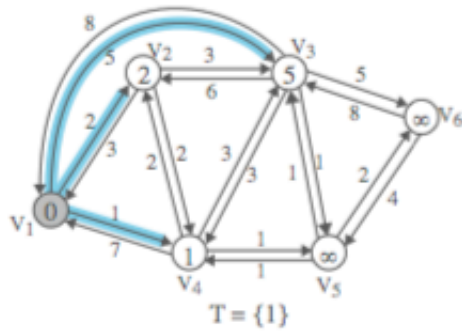
- s nodo origen del que pretendemos calcular los caminos
- $L(n_i)$ distancia mínima de s a n_i
- T conjunto de nodos a los que ya conocemos la distancia mínima desde s y el camino

Algoritmo de Dijkstra

Cada iteración son 3 pasos

- Paso 1 [Inicialización]
 - $T = \{s\}$ Conjunto de nodos calculados
 - $L(n) = w(s, n)$ para $n \neq s$
 - Las distancias iniciales a los vecinos son los pesos
 - Las distancias iniciales a los no vecinos son infinito
- Paso 2 [Buscar siguiente nodo a añadir]
 - Encontrar el vecino (que no pertenezca a T) de algún nodo de T con menor distancia a s . Le llamaremos x
 - Incorporar x a T
 - Incorporar también el enlace a x que consigue esa distancia mínima
- Paso 3 [Actualizar los nuevos costes mínimos]
 - $L(n) = \min[L(n), L(x) + w(x, n)]$ para todo $n \notin T$
 - Si cambiamos $L(n)$ por $L(x) + w(x, n)$ entonces el camino de s a n es el camino de s a x seguido por el enlace de x a n

Ejemplo



Ejemplo

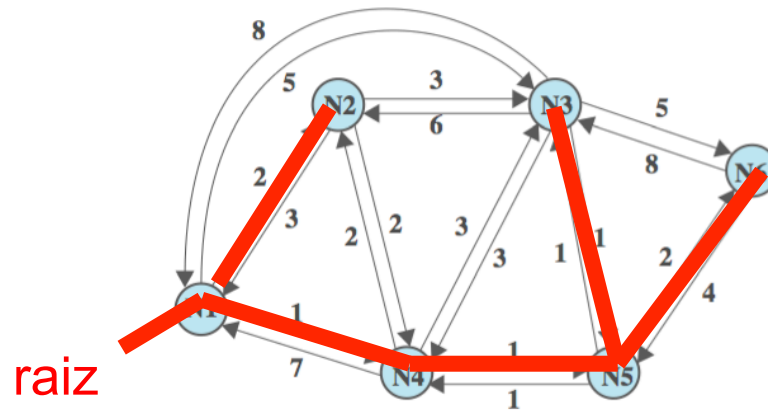
Iter	T	L(2)	Path	L(3)	Path	L(4)	Path	L(5)	Path	L(6)	Path
1	{1}	2	1-2	5	1-3	1	1-4	∞	-	∞	-
2	{1,4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
3	{1, 2, 4}	2	1-2	4	1-4-3	1	1-4	2	1-4-5	∞	-
4	{1, 2, 4, 5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
5	{1, 2, 3, 4, 5}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6
6	{1, 2, 3, 4, 5, 6}	2	1-2	3	1-4-5-3	1	1-4	2	1-4-5	4	1-4-5-6

Resumen Dijkstra

- Funciona: Calcula si existe el camino más corto
- Necesita la topología completa
 - El coste de todos los enlaces de la red
 - Si un nodo quiere usar Dijkstra para calcular el camino a cualquier nodo necesitará comunicarse con todos los demás nodos para obtener información de los enlaces
 - Que algoritmo podríamos usar para comunicarnos con todos los demás?
- Problema: si hemos calculado los caminos con Dijkstra a partir de un nodo... ¿podemos obtener los caminos desde otro nodo?

Más sobre Dijkstra

- En realidad el algoritmo de Dijkstra nos permite construir lo que se llama un árbol de expansión (spanning-tree) del grafo original.



- Abarca todos los nodos del grafo y no contiene ciclos
- El spanning-tree es centrado en un nodo y no vale para otros
- Si queremos calcular caminos minimos a otro nodo debemos volver a aplicar el algoritmo con otro nodo de origen
- El algoritmo de Dijkstra viene bien si queremos un algoritmo de enrutamiento de calculo distribuido a partir de la información de toda la red

Conclusiones

- El enrutamiento es un problema fundamental en redes de paquetes
- Tipos de enrutamiento
 - Centralizados y distribuidos
 - Estático, inundación, aleatorio, adaptativo
- Enrutamiento adaptativo basado en el mínimo coste es el más utilizado
- Algoritmos básicos
 - Algoritmos de mínimo coste
 - Algoritmo de Dijkstra
 - Permite calcular todos los caminos desde un nodo conociendo la información completa de la red
 - Algoritmo de Bellman-Ford (próximo día)
- Próxima clase:
 - Bellman-Ford
 - Problemas
 - Como se usa todo esto en Internet? (+1 semana)