

Enrutamiento

Introducción y conceptos generales

Area de Ingeniería Telemática

<http://www.tlm.unavarra.es>

Redes

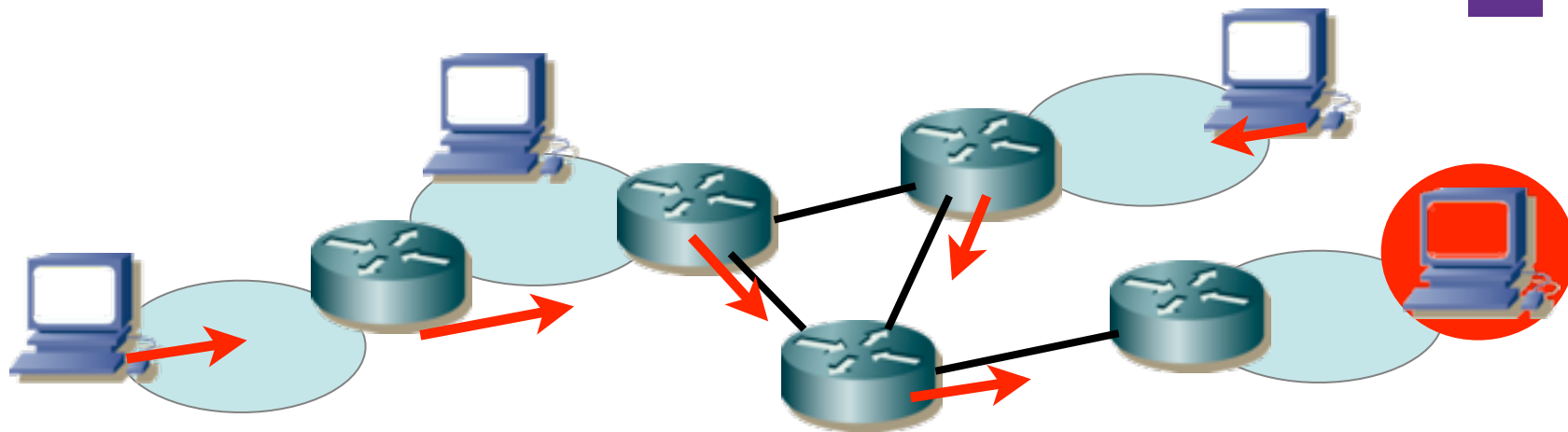
4º Ingeniería Informática

Hoy...

1. Introducción a las redes
2. Tecnologías para redes de área local
3. Conmutación de circuitos
4. Tecnologías para redes de área extensa y última milla
- 5. Encaminamiento**
 - Arquitectura de conmutadores de paquetes
 - Control de acceso al medio
 - Transporte extremo a extremo

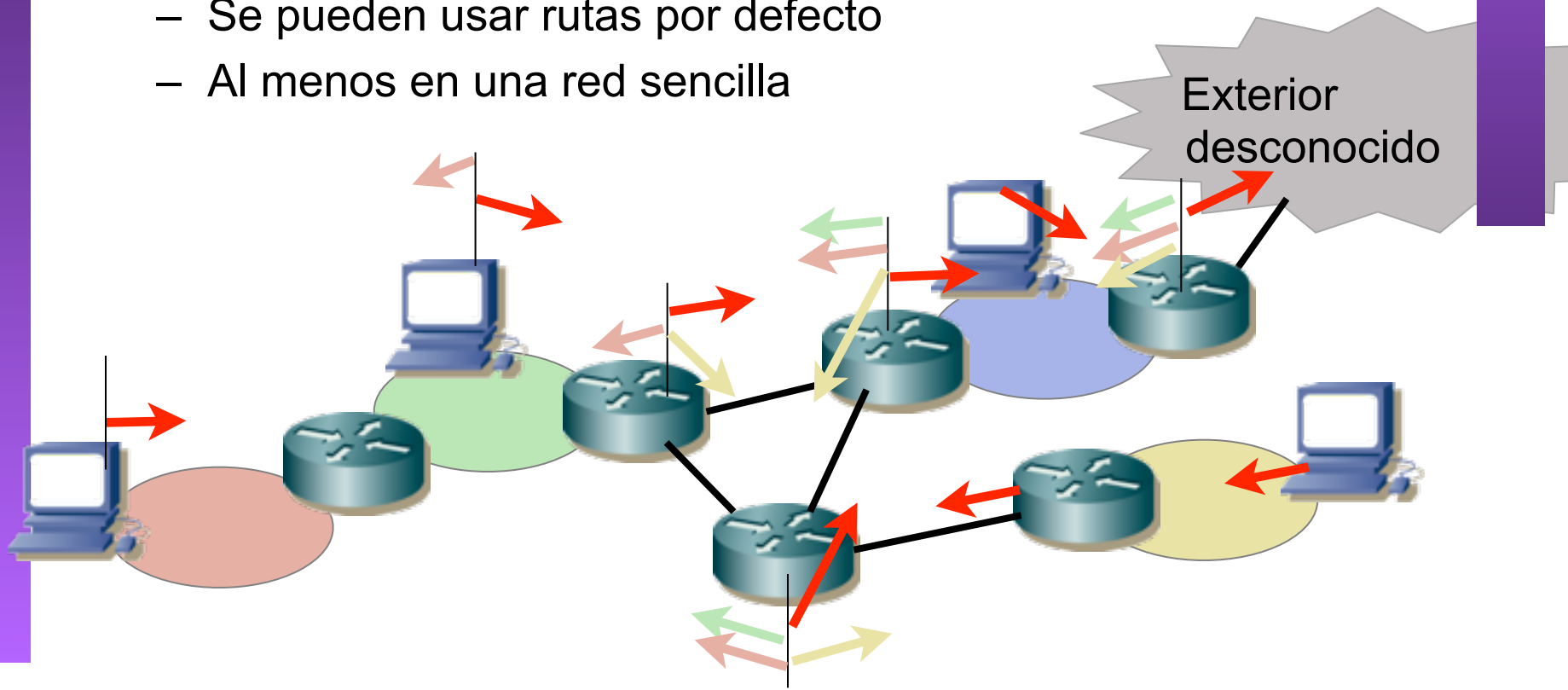
El problema del encaminamiento

- El nivel de red reenvía paquetes hacia su destino
- En IP: reglas de reenvío, vecindad indicada con IP y mascara, tablas de rutas... = Reenvío
- **Encaminamiento / enrutamiento / routing**
Cómo decidir que caminos poner en las tablas de rutas
- Es uno de los problemas fundamentales de redes



Enrutamiento estático

- Solución más fácil: let the human do it
= lo que hacíais en los problemas de Redes de Computadores.
¿En esta red cual deberían ser las tablas de rutas?
- Parece fácil de pensar con un poco de paciencia
 - Se pueden resumir redes
 - Se pueden usar rutas por defecto
 - Al menos en una red sencilla



Enrutamiento estático: problemas

- No es tan fácil cuando no hay exterior
- No es tan fácil cuando el numero de nodos crece
- No es tan fácil cuando hay caminos alternativos y ciclos
- Los administradores se equivocan
- Los administradores pertenecen a diferentes empresas que no confían entre si
- ¿Qué hacemos para poner un nuevo router, una nueva red, añadir enlaces?
- ¿Se puede hacer que el encaminamiento funcione de forma automática?
- **Encaminamiento dinámico**

Enrutamiento dinámico

- En redes de datagramas
 - Cada paquete es reenviado según la información de enrutamiento que hay en cada nodo en cada momento.
 - Si cambian las condiciones de la red entre un paquete y otro los paquetes pueden seguir distintos caminos
 - Robusto, se adapta rápido a los cambios
- En redes de circuitos/circuitos virtuales
 - Al establecer el circuito/camino se utiliza la información de enrutamiento disponible en ese momento
 - Una vez establecido el circuito se reenvía más rápido. Pero si cambian las condiciones de la red el camino ya no es el mejor
 - Las decisiones de enrutamiento son menos frecuentes (=podemos la misma capacidad en usar algoritmos mas complicados para decidir el camino)

¿Cuál es el destino?

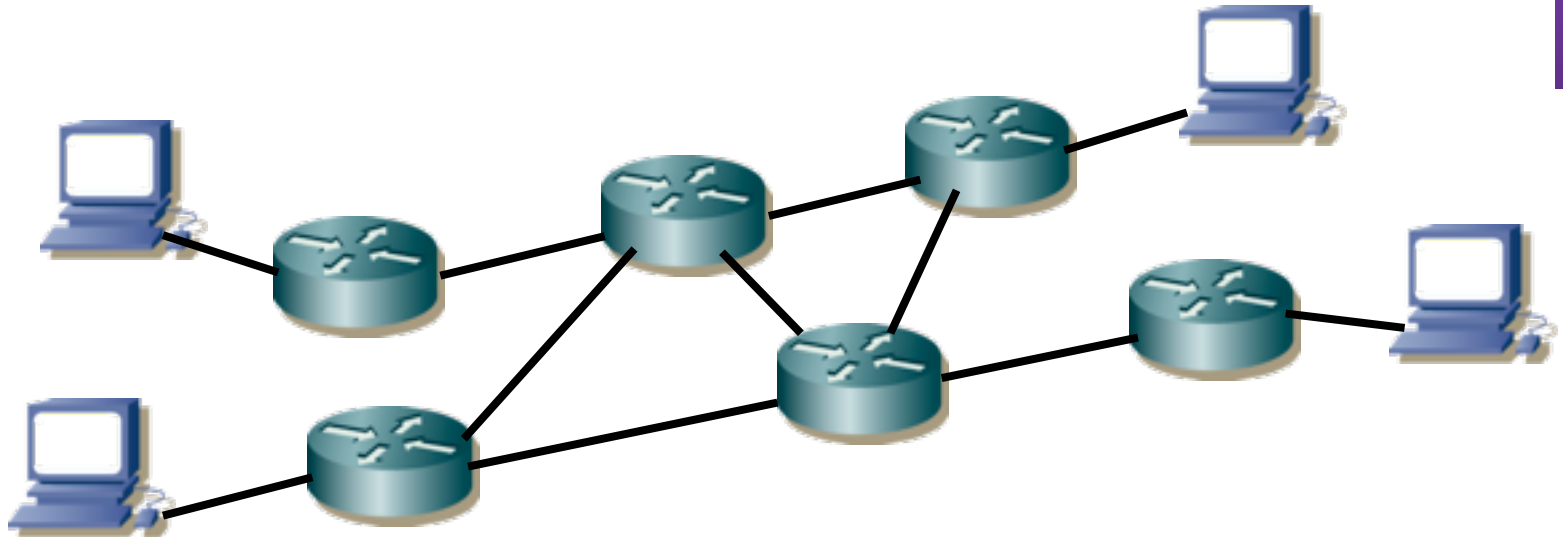
- **Unicast:** un solo destino
 - **Broadcast:** el destino son todos los nodos
 - **Multicast:** el destino es un conjunto de nodos, un mensaje tiene que llegar a todo el conjunto
 - **Anycast:** el destino es un conjunto de nodos, un mensaje tiene que llegar a uno cualquiera del conjunto
-
- Nos centraremos en unicast
 - Las herramientas/algoritmos son también la base del enrutamiento broad-, multi-, any- cast

Enrutamiento dinámico

- ¿Qué información necesitamos de la red?
 - Un mapa de la red entera?
 - Se puede hacer algo con información parcial?
 - Se puede hacer algo sin información?
- Para programar y mantener lo mejor es lo mas simple...

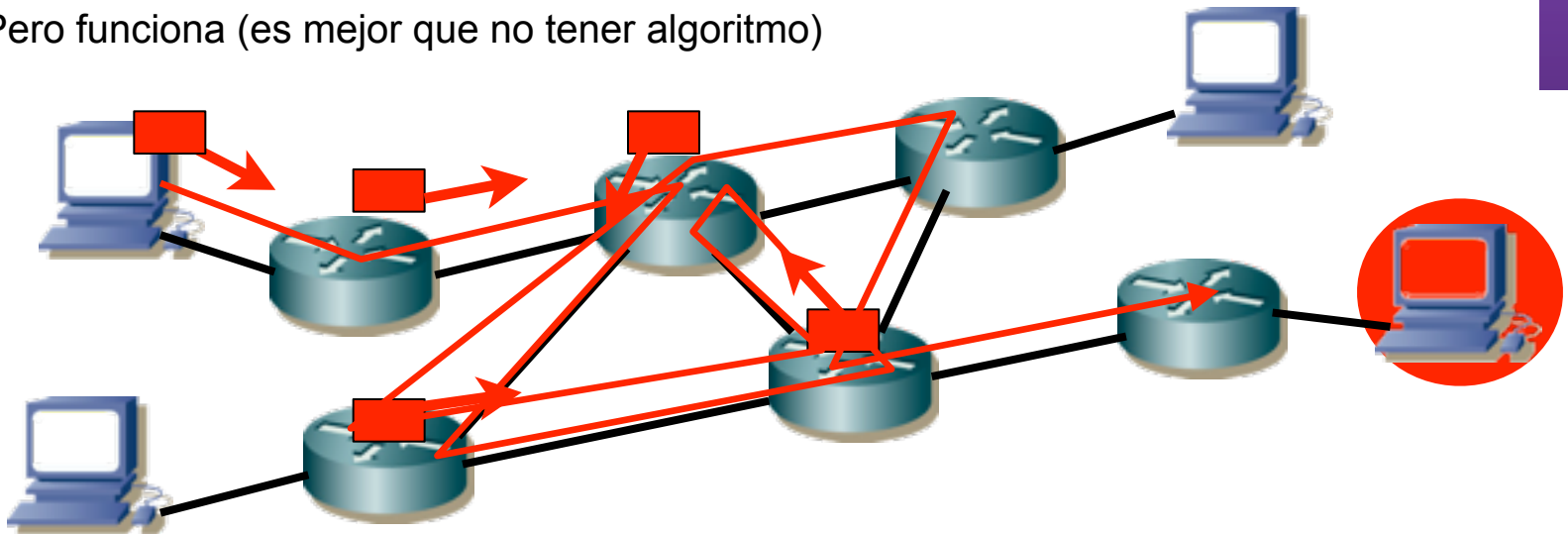
Empezando desde cero

- ¿Hay algún algoritmo simple que permita que los paquetes lleguen a su destino, sin mucha complicación?
- En realidad si
- Y con información mínima en cada router (sin tabla de rutas)



Enrutamiento aleatorio !!

- Cuando tengo que reenviar un paquete que no es para mi (podría extenderse a ni para mis redes conectadas)...
elige un **siguiente salto aleatorio** entre todos los posibles y envíasele a el
- No es muy eficiente
- El paquete puede tardar mucho en llegar pero al final llega (con TTL infinito) o bien podemos jugar con TTLs y probabilidades de entrega
- Puede pasar mas de una vez por los nodos
- Normalmente no llega por el camino mas corto
- Ni todos los paquetes que llegan van por el mismo camino =no mantiene el orden de paquetes
- Pero funciona (es mejor que no tener algoritmo)

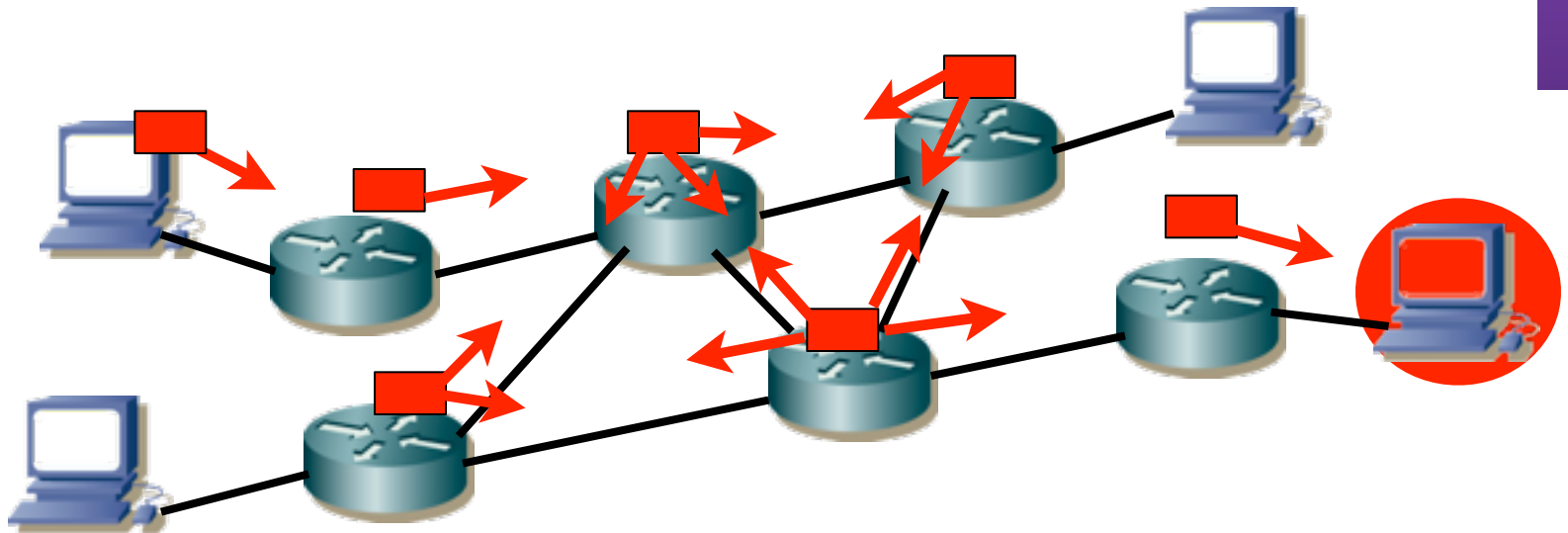


Enrutamiento por inundación

- Cuando tengo que reenviar un paquete que no es para mi (podría extenderse a ni para mis redes conectadas)...

envíalo a todos los siguientes saltos menos por el que llego

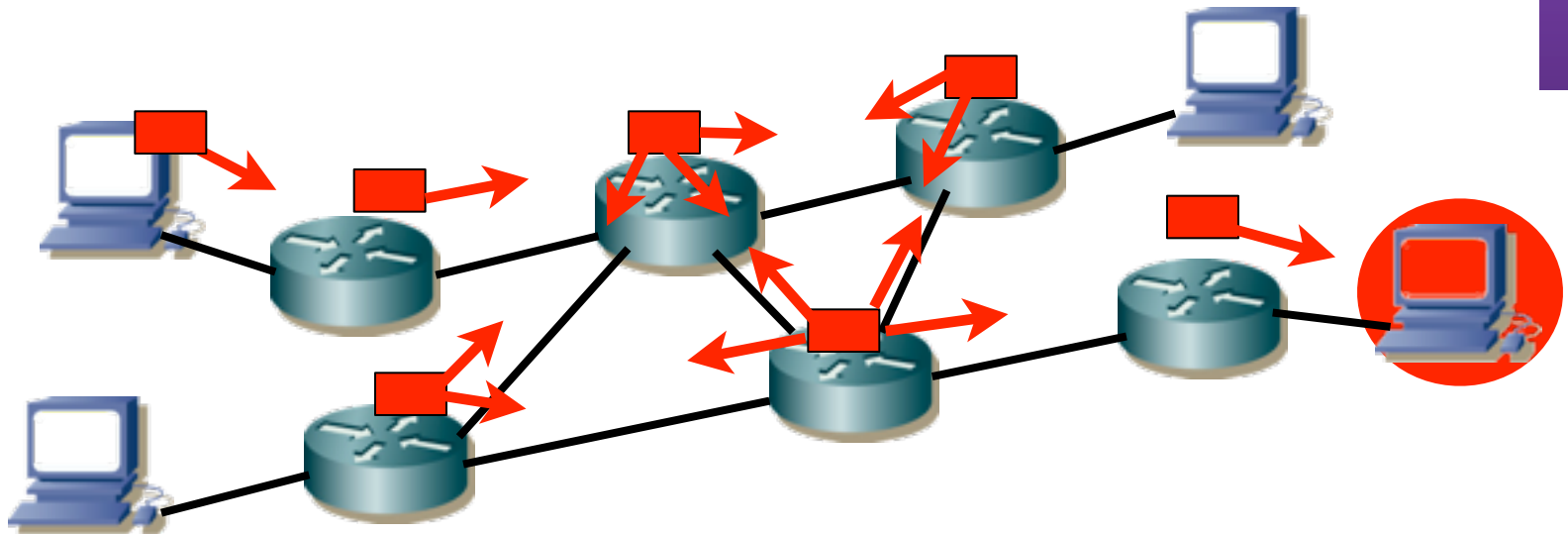
- Genera tráfico extra. Cuanto?
- El paquete llega seguro (salvo que se usen TTLs)
- El paquete llega por el camino mas corto
- El paquete llega probablemente mas de una vez. Hay que usar técnicas para garantizar que solo se entrega una vez (identidad del paquete)



Enrutamiento por inundación

- ¿Podemos resolver todos los problemas?
- El paquete llega probablemente mas de una vez. No queremos entregar duplicados
 - Campo de Identidad en el paquete. Hay que garantizar que no se use de nuevo la misma identidad en el marco de tiempo en que podría confundirse con una copia
- Si hay ciclos en el grafo la inundación genera tráfico infinito
 - TTL en los paquetes (facil, pero ya no podemos garantizar que llegue seguro)
 - No reenviar paquetes recientes

Parece facil... pero... que es un paquete reciente? Implica guardar los paquetes que hemos reenviado en los últimos T segundos. Estado en el router

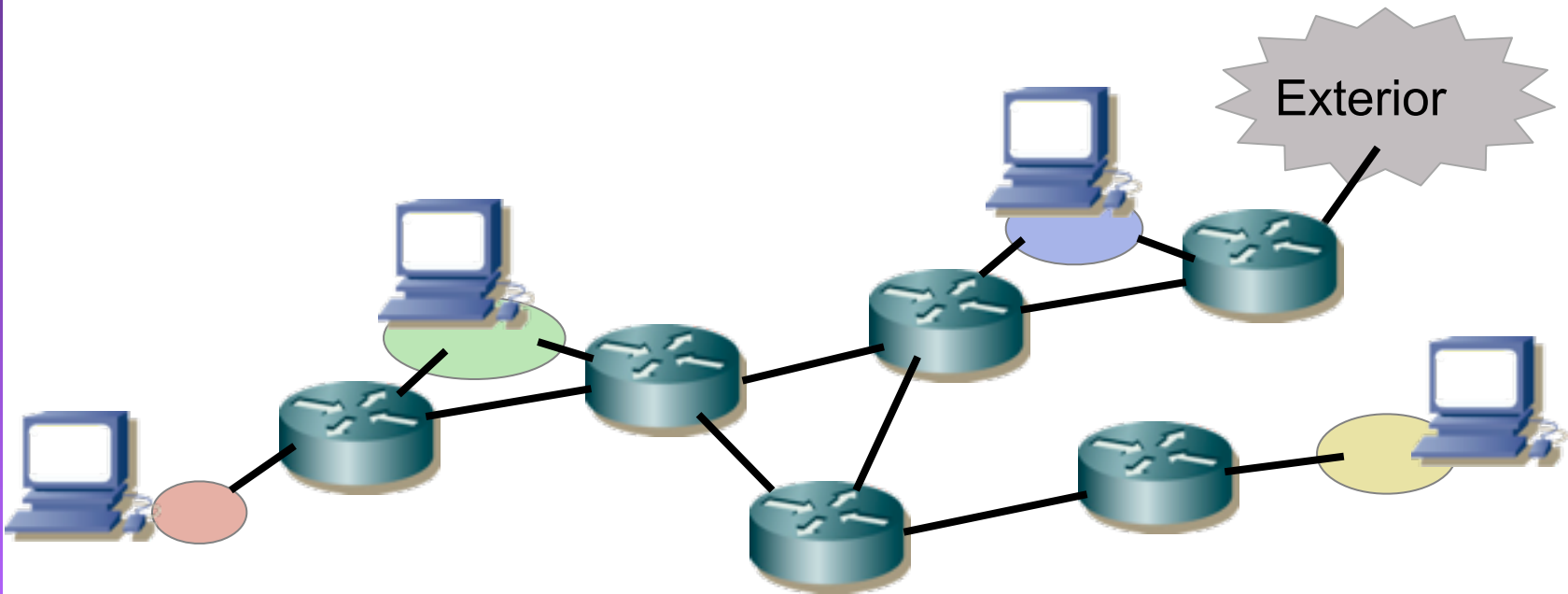


Enrutamiento sin información

- **Enrutamiento aleatorio y por inundación**
- Presentan problemas de poca eficiencia/mucho tráfico extra
- Pero son simples y funcionan
- Hay situaciones en las que tiene sentido usarlos
- Incluso hoy en día
- A veces combinados con otros sistemas de enrutamiento
 - Por ejemplo la inundación se puede controlar más fácil si estamos en un entorno en que hay otro protocolo de enrutamiento funcionando
Esto se usa para conseguir broadcast y multicast
 - Por ejemplo la inundación puede ser aceptable si solo se usa al establecer un circuito porque garantiza encontrar el camino más corto.
- Pero en general en una red tan compleja y de la extensión de Internet los protocolos de este tipo no escalan.
 - Mejor mantener los paquetes a que vayan solo por un camino hacia el destino
Mejor si es el más corto
 - Para eso hace falta utilizar información en el enrutamiento

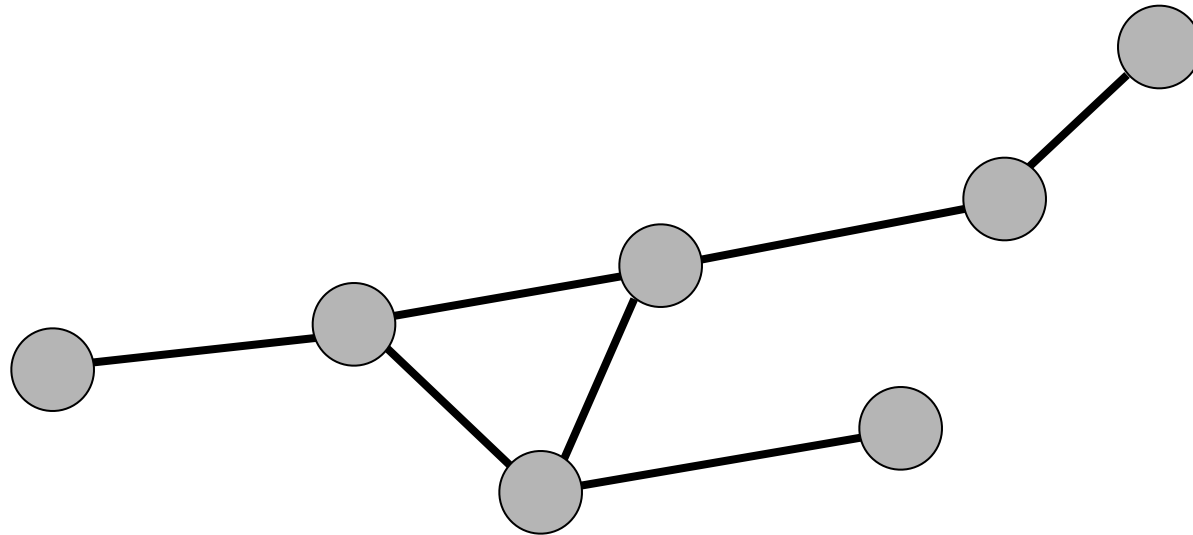
Enrutamiento con información de la red

- Simplificaremos el problema pensando que las redes son una serie de nodos interconectados entre si formando un grafo. Las redes de area local se pueden pensar simplemente como destinos
Los detalles de como se distinguen redes y routers quedan para los protocolos de enrutamiento concretos en proximas clases



Enrutamiento con información de la red

- Tenemos un grado en el que los elementos son reenviadores de paquetes: nodos
- Los enlaces son conexiones entre estos (punto a punto, LAN...)
- Los nodos son posibles orígenes y destinos
- ...



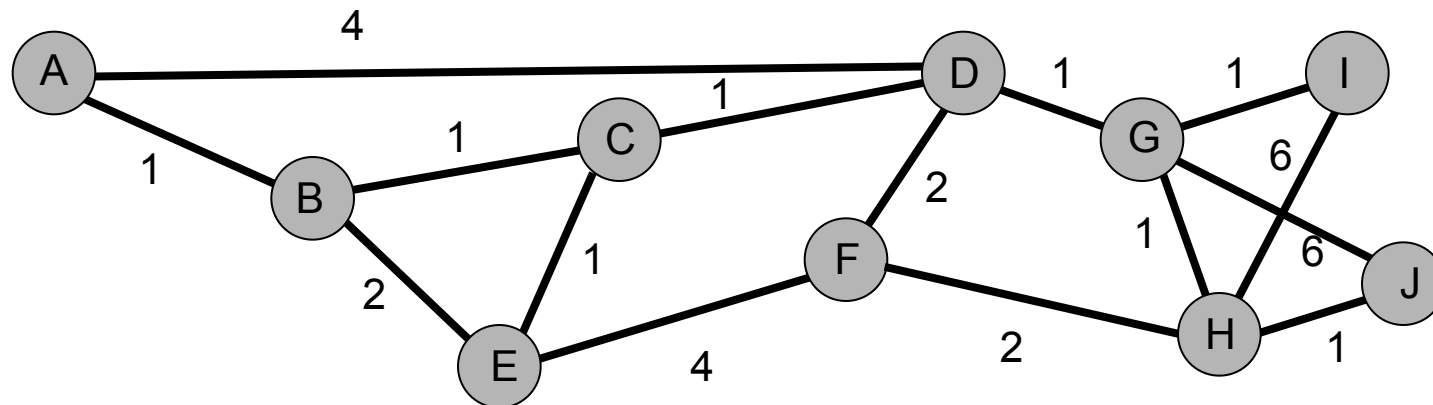
Enrutamiento con información de la red

- Los enlaces pueden ser iguales o no
- Normalmente asignamos un peso a cada enlace que mide cuanto nos cuesta o queremos evitar ese enlace comparado con otro

Se supone que el peso es aditivo

Preferimos A-B-C-D ($1+1+1$) que A-D (4)

- Si asignamos pesos 1 lo que nos interesa es el numero de saltos del camino total
- Por simplicidad suponemos un grafo no dirigido
(= los enlaces son bidireccionales)



Enrutamiento con información de la red

- La red puede cambiar
 - Los enlaces pueden crearse y destruirse
 - Los pesos de los enlaces pueden cambiar (pueden depender de la carga que este atravesando el enlace por ejemplo)
- El enrutamiento dinámico debe adaptarse a estos cambios
- Los protocolos reales se han centrado más en adaptarse a los cambios en la topología que a los cambios de la carga.
 - La carga puede cambiar muy rápido
 - Es malo que el enrutamiento cambie demasiado rápido

Compromiso robusto vs estable

- Cambiar rápido: robusto, ante una caída reacciona rapido encontrando nuevos caminos
- En los periodos de cambio pueden generarse situaciones anómalas, bucles de enrutamiento
- Resistencia a cambiar: sistema más estable y condiciones mas predecibles

Clasificación

- Distribuido vs Centralizado
- Centralizado
 - Un ente recopila toda la información de la red y decide las rutas
 - Este ente puede tomar decisiones conociendo toda la red
- Distribuido
 - Cada nodo se comunica con el resto de nodos y utiliza la información que obtiene para calcular su tabla de rutas
 - Los nodos pueden calcular la tabla de rutas con información parcial o conseguir toda la información

Clasificación

- Proactivo vs reactivo
- Enrutamiento **proactivo**
 - Los protocolos de enrutamiento funcionan de forma continua rellenando las tablas de rutas para todos los caminos y cambiandolas cada vez que hay un cambio en la topología
 - Al enviar un paquete a un destino ya existe la entrada en la tabla
 - Este es el caso normal en Internet
- Enrutamiento **reactivo**
 - Las tablas de rutas se calculan solo para los destinos que se están usando
 - El calculo se dispara cuando hace falta enviar a un destino
 - Asociado a redes de circuitos
 - O bien a redes donde se intenta evitar enviar información (i.e. inalámbricas)
 - Por ejemplo:
 - red de circuitos virtuales sin tablas de rutas.
 - cuando se va a establecer circuito a un destino que no es vecino
 - enviamos inundación preguntando por el destino
 - cada nodo que reenvía la información añade su dirección
 - por lo que el paquete que recibe el destino tiene el camino

Características avanzadas

- Confianza/seguridad
- Broadcast y multicast
- Un solo camino o caminos paralelos
- Con calidad de servicio
- Usando información geográfica

El modelo hasta ahora

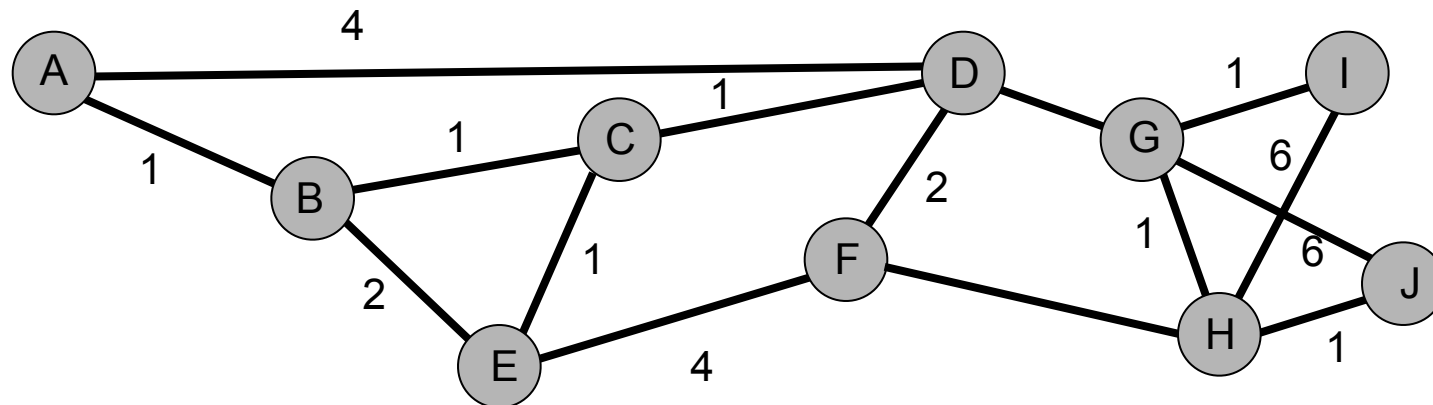
- Tenemos un grafo que representa nuestra red
- Supongamos un sistema de enrutamiento...

centralizado, proactivo, adaptativo que debe buscar caminos unicast desde todos los nodos a todos los nodos para que estén calculadas todas tablas de rutas. Se asume que la topología no va a variar en un tiempo prudencial y los pesos de los enlaces tampoco...

Supongamos todos los caminos a un destino desde todos los nodos y repetir con todos los nodos como destinos

- Sabríais programar eso?

La respuesta debería ser SI



Solución

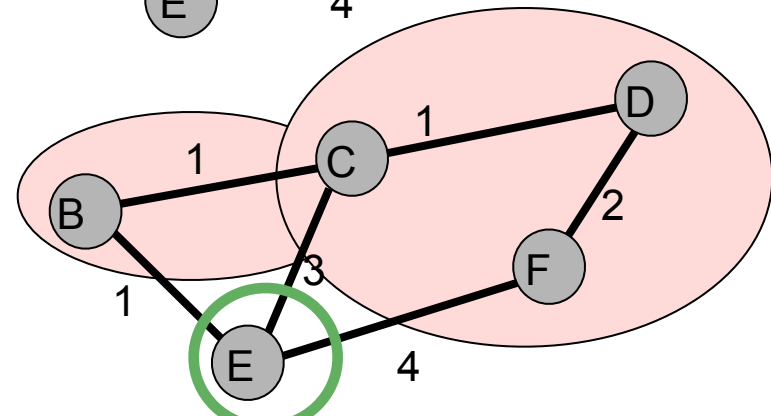
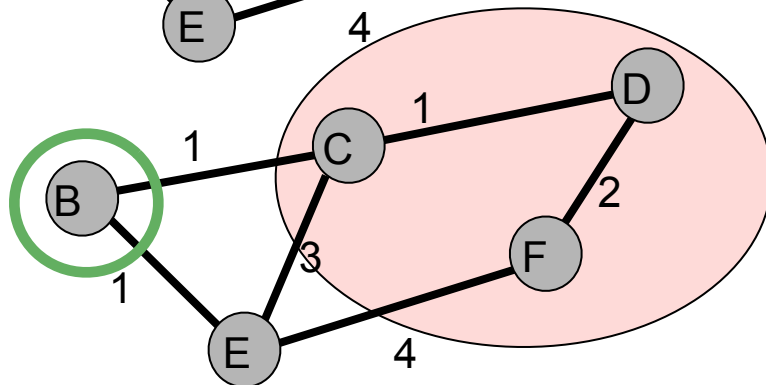
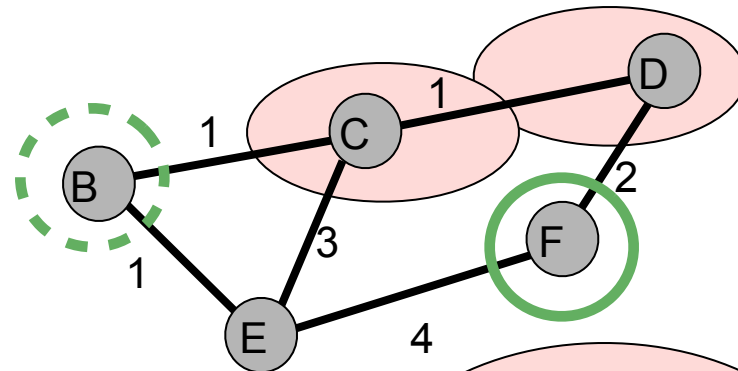
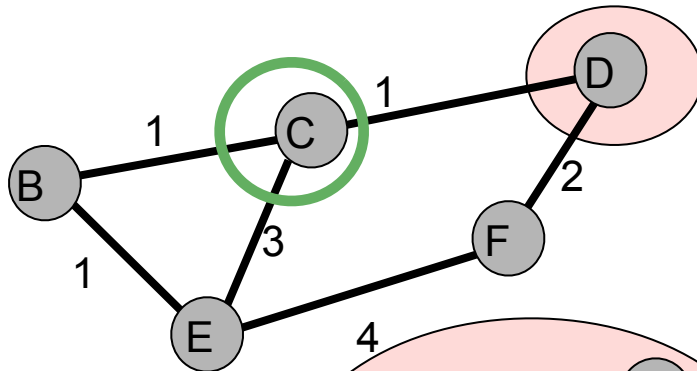
- Si tienes toda la información de la red
- Hay algoritmos de encontrar caminos en grafos
i.e. Dijkstra, Bellman-Ford, A*
- Calculas los caminos de todos los nodos al nodo S
- Sink-tree / spanning tree con root S
- Rellenas la tabla de rutas usando esa información

Algoritmos para encontrar caminos

- Algoritmo de **Dijkstra** para la distancia mínima
- Conocemos: Nodos i Nodo raíz r pesos $w(i,j)$
- Mantenemos: $d(i)$ mejor distancia de r al nodo i conocida hasta ahora
 T conjunto de nodos a los que ya conocemos la distancia mas corta
 $d(i)$ es la distancia menor de r hasta el nodo i pasando solo por nodos que están en T
- Algoritmo:
 Inicializar
 $d(i)=\text{infinito}$ $d(r)=0$ si el nodo i es vecino de r [$w(i,r)<\text{infinito}$] $d(i)=w(i,r)$
 $T=\{r\}$
 Mientras haya nodos que no pertenezcan a T
 elegir el nodo i que no este en T con menor $d(i)$
 añadir el nodo i a T
 actualizar $d(k)$ de los nodos vecinos al nodo i que no están en T .
 Si $d(i)+w(k,i) < d(k)$ entonces $d(k)=d(i)+w(k,i)$
 [es menor la distancia pasando por i que la que ya tenia]

Dijkstra ejemplo

T	d(B)	d(C)	d(D)	d(E)	d(F)
{D}	infinito	1	0	infinito	2
{D,C}	2	1	0	4	2
{D,C,F}	2	1	0	4	2
{D,C,F,B}	2	1	0	3	2
{D,C,F,B,E}	2	1	0	3	2



Manteniendo el camino

- Usando el algoritmo de **Dijkstra** para el camino mínimo
- Conocemos: Nodos i Nodo raiz r pesos $w(i,j)$
- Mantenemos: $d(i)$ mejor distancia de r al nodo i conocida hasta ahora
 $s(i)$ siguiente nodo a i en el camino hacia r
 T conjunto de nodos a los que ya conocemos la distancia mas corta

Algoritmo:

Inicializar

$d(i)=\text{infinito}$ $d(r)=0$ si el nodo i es vecino de r [$w(i,r)<\text{infinito}$] $d(i)=w(i,r)$

$T=\{r\}$

$s(i)=\text{desconocido}$ si i es vecino de r [$w(i,r)<\text{infinito}$] $s(i)=r$

Mientras haya nodos que no pertenezcan a T

 elegir el nodo i que no este en T con menor $d(i)$

 añadir el nodo i a T

 actualizar $d(k)$ de los nodos vecinos al nodo i que no están en T .

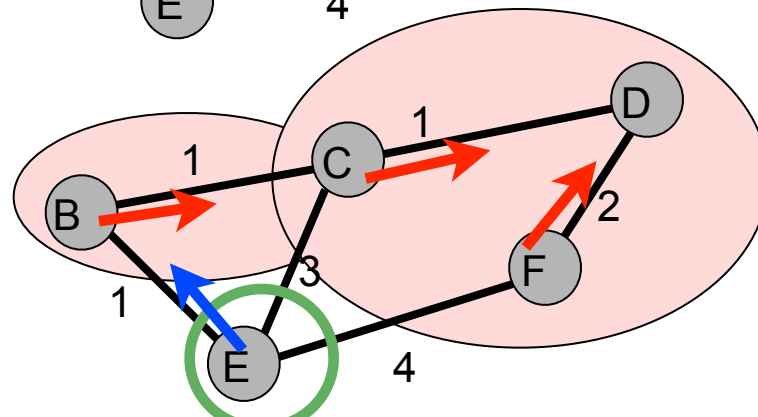
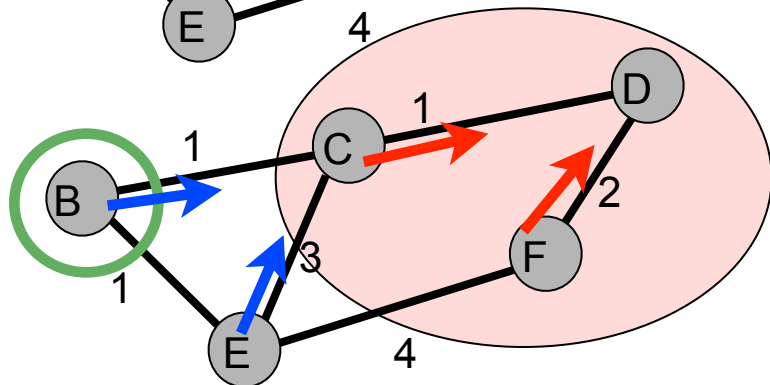
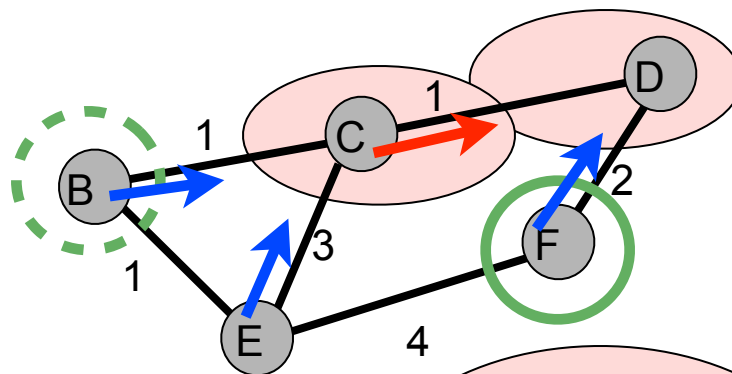
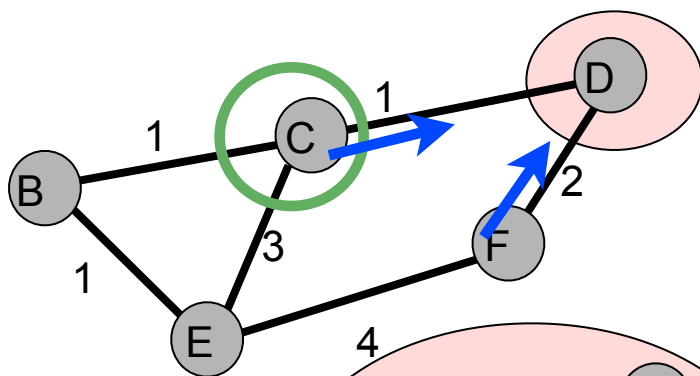
 Si $d(i)+w(k,i) < d(k)$ entonces $d(k)=d(i)+w(k,i)$

 y tambien $s(k)=i$

 [si el camino por i es mejor i es el nuevo siguiente salto de k]

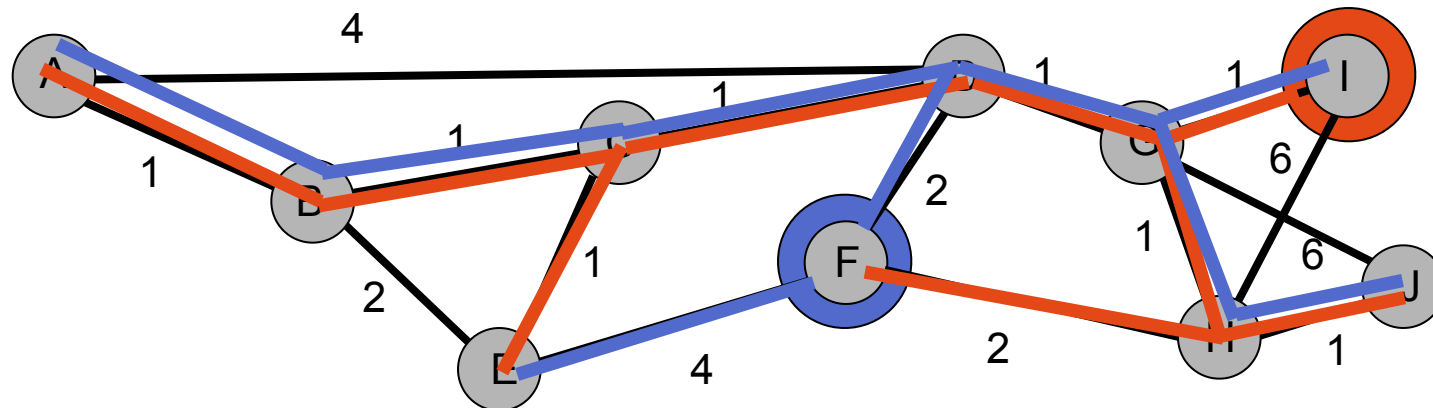
Dijkstra ejemplo con camino

T	d(B) / s(B)	d(C) / s(C)	d(D) / s(D)	d(E) / s(E)	d(F) / s(F)
{D}	infinito / desc	1 / D	0 / soy yo	infinito / desc	2 / D
{D,C}	2 / C	1 / D	0 / soy yo	4 / C	2 / D
{D,C,F}	2 / C	1 / D	0 / soy yo	4 / C	2 / D
{D,C,F,B}	2 / C	1 / D	0 / soy yo	3 / B	2 / D
{D,C,F,B,E}	2 / C	1 / D	0 / soy yo	3 / B	2 / D



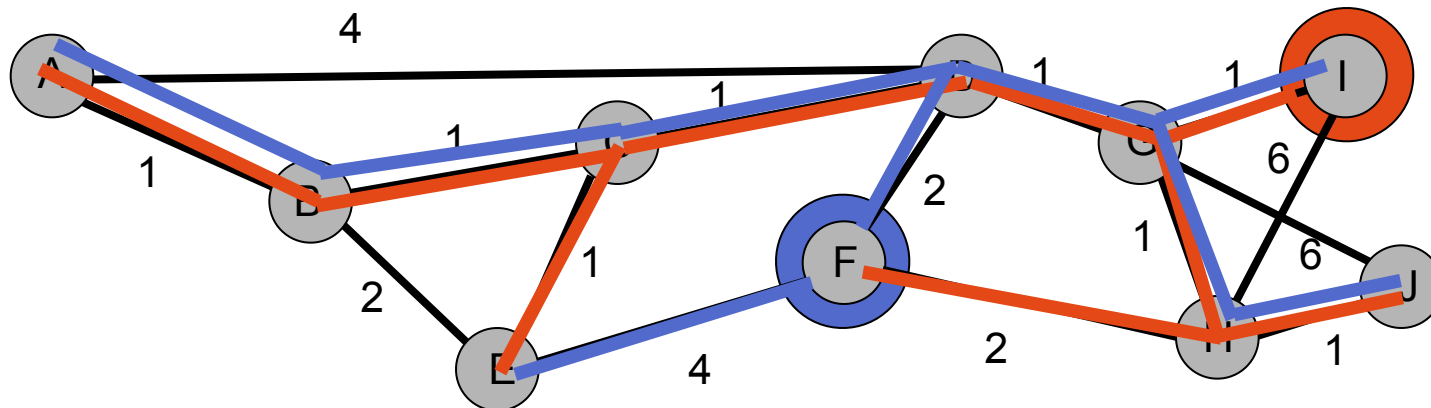
El resultado...

- Todos los caminos a un destino forman un árbol de expansión de coste mínimo del grafo
- Los caminos a cada destino son un árbol pero no tienen por que coincidir sus enlaces
- Con esto rellenaríamos la tabla de rutas de cada nodo



Adaptativo

- Como hacemos para que el enrutamiento centralizado se adapte a cambios en la red?
 - Cambio en los pesos
 - Cambio en la existencia de nodos o enlaces
- Si un ente central tiene toda la información: trivial
 - Volver a calcular los arboles con el nuevo grafo cuando se detectan cambios
 - Periodicamente obtener toda la información de la red y calcular las tablas de rutas



Distribuido

- Como hacemos para hacer el calculo de forma distribuida
 - Se puede hacer un algoritmo para que cada nodo calcule la parte del arbol que le interesa (quien es el siguiente salto) sin generar el resto del arbol
 - SI, pero no con el algoritmo de Dijkstra
 - Ademas se adapta naturalmente a los cambios de la red
 - Pero tiene algunos problemas
 - O bien podemos generar todo el arbol en cada nodo
- En cualquier caso cada nodo necesita recibir información de otros nodos... al menos de sus vecinos
- En esto se basa el enrutamiento de Internet y los protocolos clásicos
 - Algoritmos de calculo de rutas con más o menos información
 - Protocolos para comunicar información de enrutamiento a otros nodos
 - Reciben el nombre genérico de **protocolos de enrutamiento** i.e. RIP, OSPF, IS-IS, EGP, BGP ...

Temas avanzados

- Como hacemos para hacer el calculo de los arboles...
 - Para que puedan usarse también para multicast, broadcast, anycast...
 - De forma que sea resistente a que algunos nodos funcionen mal y envíen información incorrecta... o bien que algunos nodos quieran atacar la red y funcionen deliberadamente mal
 - En un entorno de routing reactivo (normalmente inalámbrico o peer-to-peer)
 - En un entorno en el que los parametros de red/topologia cambien muy rapidamente
 - Que calculen mas de un camino al mismo destino, para garantizar redundancia, calidad de servicio...
 - ...

Conclusiones

- Enrutamiento: problema fundamental de redes
- Herramientas para resolverlo
 - Algoritmos de enrutamiento sin información (inundación, aleatorio...)
 - Algoritmos de grafos y árboles
 - Tipos de algoritmos
 - Internet funciona gracias a los algoritmos proactivos adaptativos distribuidos
 - Cuales son? Como funcionan? Cuales son los protocolos de enrutamiento que se usan en la realidad?
- Más en la próxima clase