RÉSEAUX – INTRODUCTION À LA FONCTION DE ROUTAGE

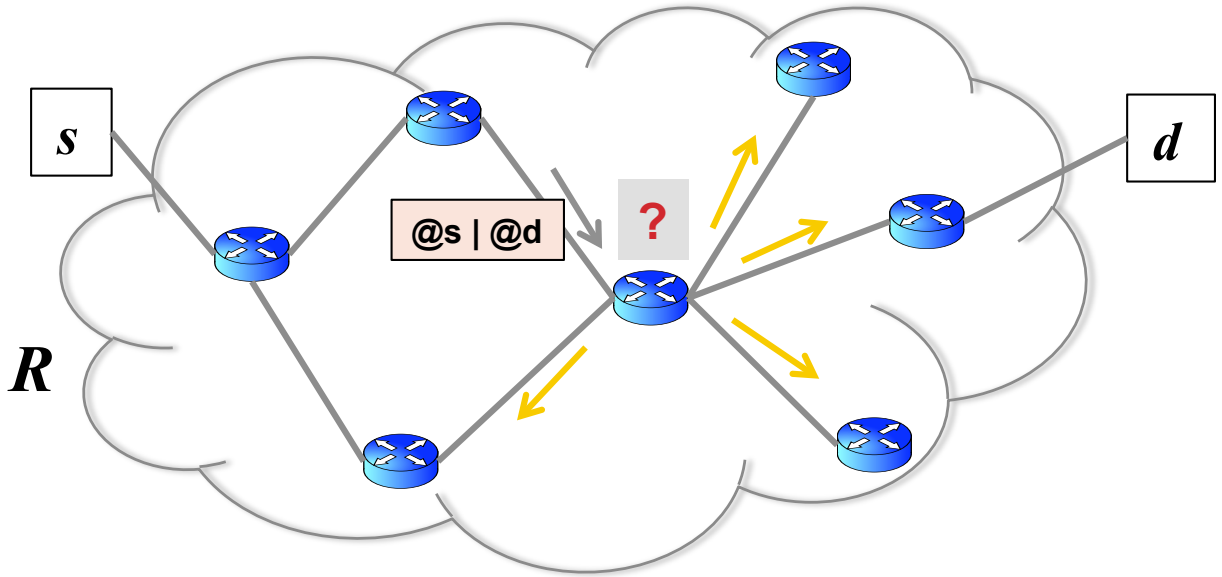
EMMANUEL LAVINAL

FORMATION ISN – MAI 2015

INTRODUCTION ROUTAGE

Service de la couche « réseau » (niveau 3)

- Tous les nœuds d'un réseau ***R*** permettent l'acheminement d'un paquet émis par une source quelconque ***s*** de ***R*** vers une destination quelconque ***d*** de ***R***.



INTRODUCTION ROUTAGE

Algorithme de routage

- Composante de la couche réseau qui a la responsabilité de sélectionner le chemin par lequel un paquet doit transiter
- Les informations qui décrivent les chemins entre les différents nœuds d'un réseau sont structurées dans une **table de routage**

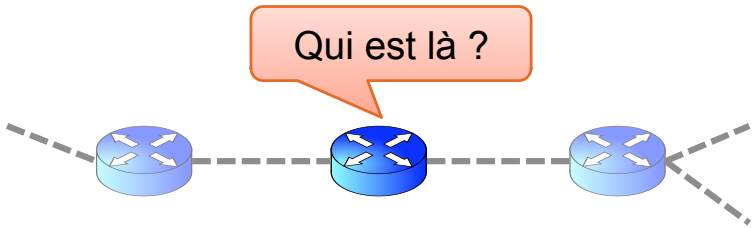
Fonction de routage (*routing*)

- **Fonction qui construit la table de routage à partir des informations échangées entre les nœuds de manière à optimiser le chemin pour transporter les paquets**
- **Note :** ne pas confondre avec la fonction de relayage ou d'expédition (*forwarding*) : pour chaque paquet reçu, elle consulte la table de routage pour savoir vers quel nœud suivant il faut expédier le paquet

Règles associées aux algorithmes de routage

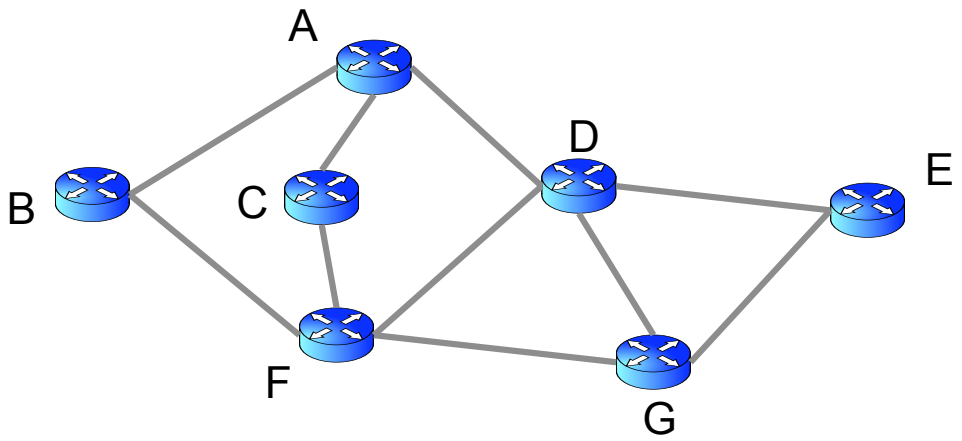
Environnement décentralisé et distribué

- Tous les nœuds ont un rôle similaire (pas de contrôleur)
- Les algorithmes s'exécutent en parallèle sur tous les nœuds
- Les nœuds obtiennent des informations de leurs voisins uniquement
- Il peut se produire des pannes de lien / nœud / message



CONCEPT DE « MEILLEUR » CHEMIN

Meilleur chemin pour aller de A à G ?



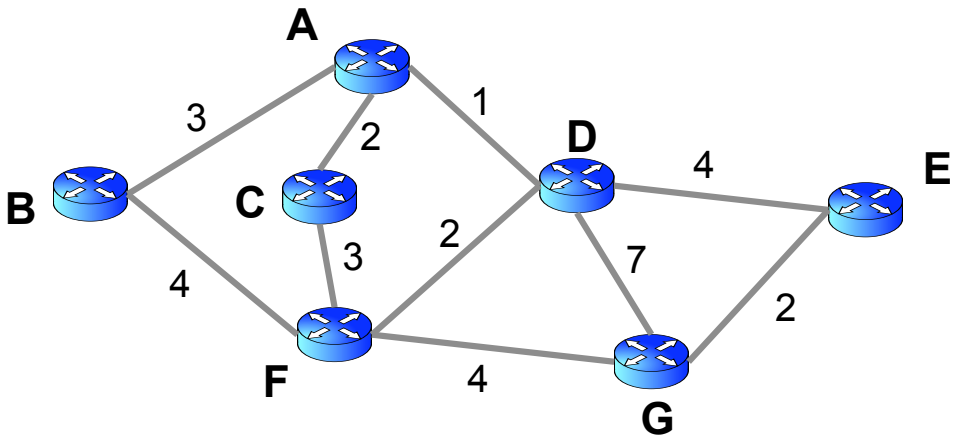
Exemples de critères

- Latence (éviter des chemins plus longs)
- Bande passante (éviter des liens lents)
- Nombre de sauts (réduire les relais)
- Coût financier

CONCEPT DE « MEILLEUR » CHEMIN

Approximation de « meilleur » par une fonction de coût

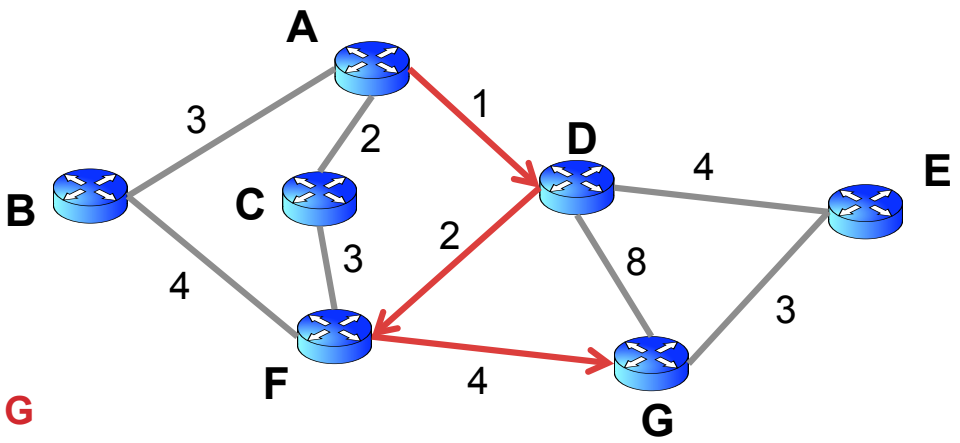
1. Affectation d'un coût (~ distance) à chaque lien
2. Définir le meilleur chemin (ou « plus court ») entre chaque couple de nœuds comme le chemin qui a un coût total minimal
3. Choix aléatoire en cas d'égalité



CONCEPT DE « MEILLEUR » CHEMIN

Approximation de « meilleur » par une fonction de coût

1. Affectation d'un coût (~ distance) à chaque lien
2. Définir le meilleur chemin (ou « plus court ») entre chaque couple de nœuds comme le chemin qui a un coût total minimal
3. Choix aléatoire en cas d'égalité



Meilleur chemin pour aller de A à G

dist(ADFG) = 1 + 2 + 4 = 7

ALGORITHMES DE ROUTAGE

Routage fixe (ou statique)

- La route à emprunter pour aller du nœud *i* au nœud *j* est calculée par avance et mise en place sur tous les routeurs à l'initialisation du réseau.
- Routage non adaptatif: les décisions de routage s'effectuent sans considération des changements de topologie, ni du trafic courant

Routage par inondation (*flooding routing*)

- Chaque paquet entrant est émis sur chaque ligne de sortie exceptée la ligne d'arrivée
- L'inondation génère un très grand nombre de paquets dupliqués (mesures palliatives : compteur de sauts, paquets numérotés,…)
- Routage non adaptatif

ALGORITHMES DE ROUTAGE ADAPTATIFS

Principe

- Les décisions de routage sont prises de façon dynamique et évoluent au cours du temps (pour prendre en compte par exemple les variations de topologie ou de trafic).
- L'implémentation d'un algorithme de routage adaptatif conduit à la définition d'un protocole de routage qui spécifie l'algorithme lui-même et le format des informations de routage échangées entre les nœuds.

Routage adaptatif distribué

- Routage par « vecteur de distance » (*distance vector*)
- Routage par « état de lien » (*link state*)

ROUTAGE PAR VECTEUR DE DISTANCE

Technique issue de l'algorithme Bellman-Ford

Principe

- Chaque nœud conserve la valeur de la distance entre lui-même et chaque destination possible : c'est le vecteur distance
- Ce vecteur de distance est calculé à partir des vecteurs distances des nœuds voisins

Table de routage

Destination	Coût	Nœud voisin
P	2	B
Q	3	B
R	2	C

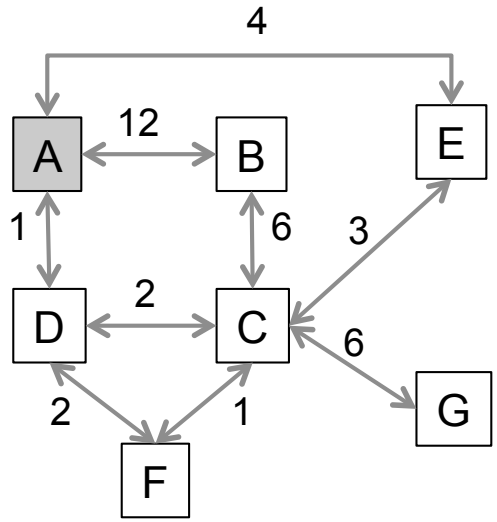
Vecteur de distance

Routage par vecteur de distance

Algorithme

- **Initialement**
  - Chaque routeur constitue un vecteur distance comportant la valeur 0 pour lui-même
  - Il diffuse ce vecteur distance sur chacune de ses liaisons
  - Chaque routeur connaît également le coût de chaque liaison adjacente
- **En opération**
  - Chaque routeur qui reçoit, de l'un de ses voisins, un vecteur distance sur une liaison L ajoute à chacune des valeurs de ce vecteur le coût de la liaison L
  - Il compare le résultat obtenu avec celles de son propre vecteur distance et éventuellement met à jour ce dernier en y rajoutant une destination nouvellement apparue ou en minimisant la distance d'une destination déjà connue
  - Si une modification a été opérée, il diffuse immédiatement son nouveau vecteur distance à chacun de ses voisins.

EXEMPLE ROUTAGE PAR VECTEUR DE DISTANCE



Initialement (sur A)

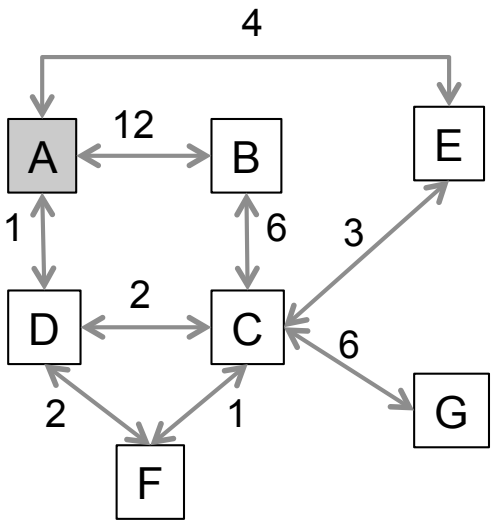
- Vecteur distance :

Destination	Coût
A	0

- Liaisons adjacentes
  - B, coût 12
  - D, coût 1
  - E, coût 4

EXEMPLE ROUTAGE PAR VECTEUR DE DISTANCE

1er échange; meilleures routes de 1 saut



Vecteurs reçus de B, D et E

	V(B)	V(D)	V(E)
A	∞	∞	∞
B	0	∞	∞
C	∞	∞	∞
D	∞	0	∞
E	∞	∞	0
F	∞	∞	∞
G	∞	∞	∞

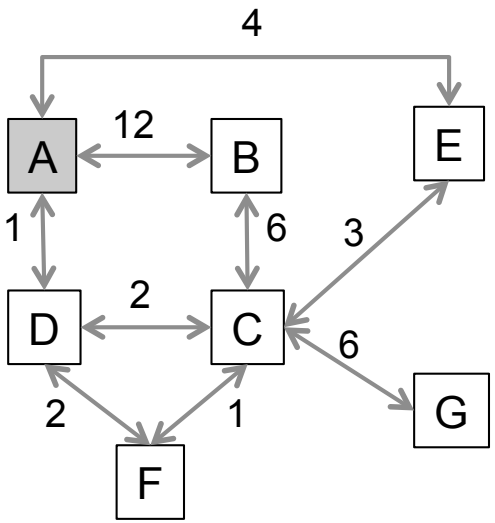
B+12
D+1
E+4

Table A

D	C	Via
A	0	-
B	12	B
C	∞	-
D	1	D
E	4	E
F	∞	-
G	∞	-

EXEMPLE ROUTAGE PAR VECTEUR DE DISTANCE

2ème échange; meilleures routes de 2 sauts



Vecteurs reçus de B, D et E

	V(B)	V(D)	V(E)
A	12	1	4
B	0	∞	∞
C	6	2	3
D	∞	0	∞
E	∞	∞	0
F	∞	2	∞
G	∞	∞	∞

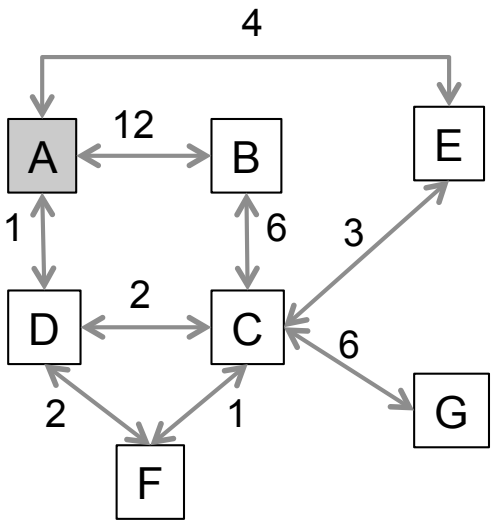
B+12
D+1
E+4

Table A

D	C	Via
A	0	-
B	12	B
C	3	D
D	1	D
E	4	E
F	3	D
G	∞	-

EXEMPLE ROUTAGE PAR VECTEUR DE DISTANCE

3ème échange; meilleures routes de 3 sauts



Vecteurs reçus de B, D et E

	V(B)	V(D)	V(E)
A	12	1	4
B	0	8	9
C	6	2	3
D	8	0	5
E	9	5	0
F	7	2	4
G	12	8	9

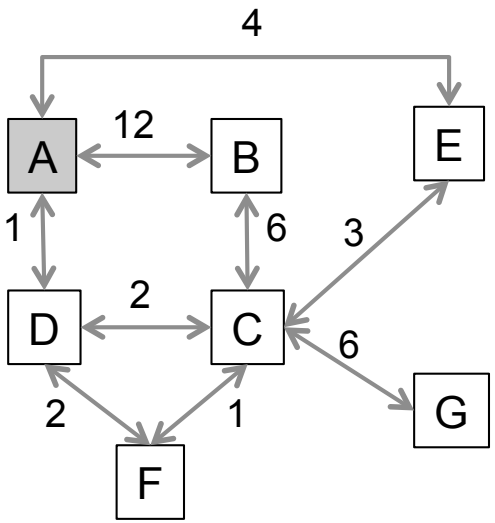
B+12
D+1
E+4

Table A

D	C	Via
A	0	-
B	9	D
C	3	D
D	1	D
E	4	E
F	3	D
G	9	D

EXEMPLE ROUTAGE PAR VECTEUR DE DISTANCE

Echanges ultérieurs (convergence)



Vecteurs reçus de B, D et E

	V(B)	V(D)	V(E)
A	9	1	4
B	0	8	9
C	6	2	3
D	8	0	5
E	9	5	0
F	7	2	4
G	12	8	9

B+12
D+1
E+4

Table A

D	C	Via
A	0	-
B	9	D
C	3	D
D	1	D
E	4	E
F	3	D
G	9	D

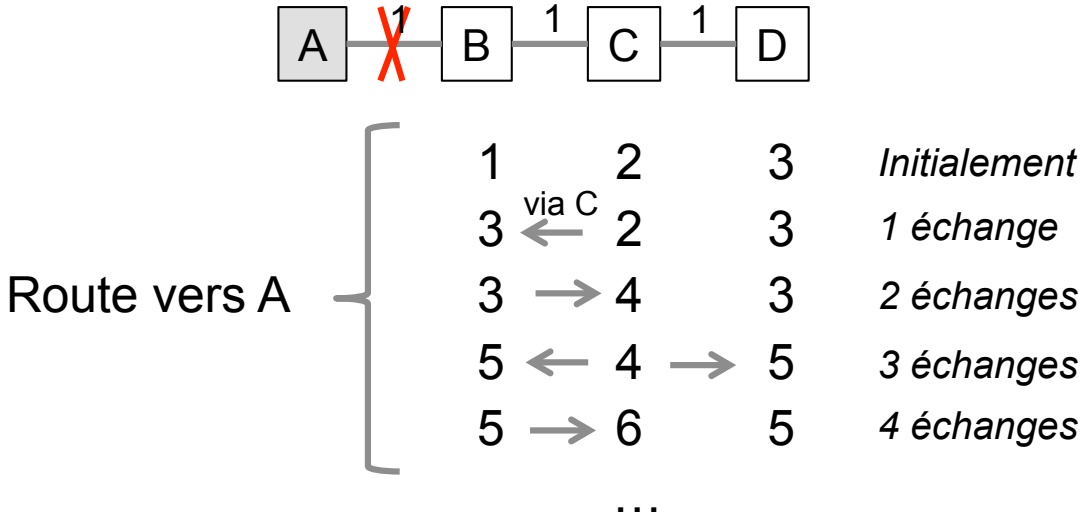
GESTION DE LA DYNAMIQUE

Ajout d'un lien ou nœud

- La nouvelle se propage à la vitesse des « sauts » des vecteurs

Suppression (ou panne) d'un nœud

- Plus de VD de ce nœud et mise à jour progressive des voisins
- Mais attention au problème du « count to infinity » !



(certaines solutions existent, non détaillées ici…)

Routage par état de lien

Principe

- Chaque nœud construit et maintient une copie complète de la carte du réseau et calcule localement les meilleurs chemins par l'algorithme de Dijkstra.

Algorithme

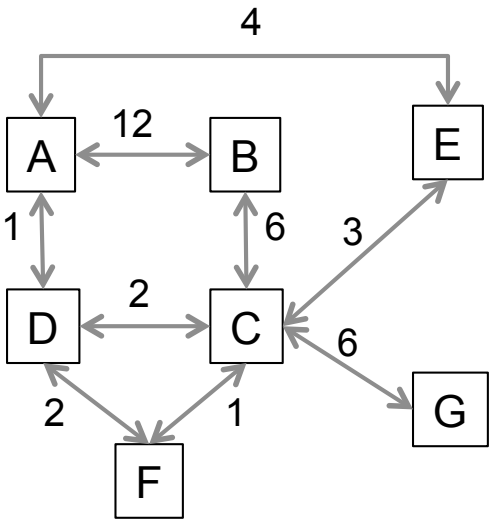
- Chaque routeur construit un paquet connu sous le nom de « **paquet d'état de liaison** » ou **LSP** (*Link State Packet*) contenant une liste de noms des voisins repérés et des coûts des liaisons respectives.
- Le LSP est transmis à tous les autres routeurs (par inondation sélective), et chaque routeur enregistre le LSP généré le plus récemment par chaque autre routeur.
- Chaque routeur, qui possède désormais une connaissance complète de la topologie du réseau et des coûts (via les LSP), calcule les routes les plus courtes vers chaque destination par Dijkstra.

ALGORITHME DE DIJKSTRA

```
G : graphe (A, S) (A : ensemble des arcs. S : ensemble des sommets)
s : nœud source.
w(i,j) : poids de l'arc entre les sommets i et j.
Pred(u) : prédécesseur de u sur le chemin.
L(u): poids du chemin de s à u (~ distance vers u)

/* Initialisation */
Pour chaque sommet v de G faire
  L(v) := +∞ ; Pred(v) := Nil ;
FinPour
L(s) := 0 ; /* distance vers la source nulle */
E := Ø ; /* sommets parcourus */
F := S ; /* tous les sommets */

/* Phase d'exploration des sommets */
Tant que F ≠ Ø faire /* TQ il reste des sommets non vus */
  u := Extraire-Min(F) ; /* u | L[u] = min{L[y], ∀y ∈ F} et retire u de F */
  E := E ∪ {u} ; /* Marque u */
  Pour chaque arc(u,v) de G faire
    Si L(v) > L(u) + w(u,v) alors
      L(v) := L(u) + w(u,v) ;
      Pred(v) := u ;
    FinSi
  FinPour
FinTantQue

 /* A la fin de l'exécution de l'algorithme :
    - L(v) contient le coût d'un chemin min pour aller de la source s à v
    - Pred(v) contient le nœud précédent v sur un tel chemin */
```

L(v) = min(L(v), L(u) + w(u,v))

EXEMPLE ROUTAGE PAR ÉTAT DE LIEN



1. Construction (et diffusion) des paquets d'état de liaison (LSP) :

A	
B	12
D	1
E	4

B	
A	12
C	6

C	
B	6
D	2
E	3
F	1
G	6

D	
A	1
C	2
F	2

E	
A	4
C	3

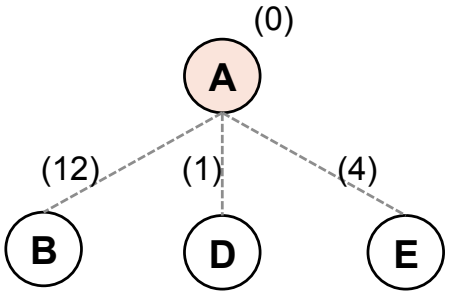
F	
C	1
D	2

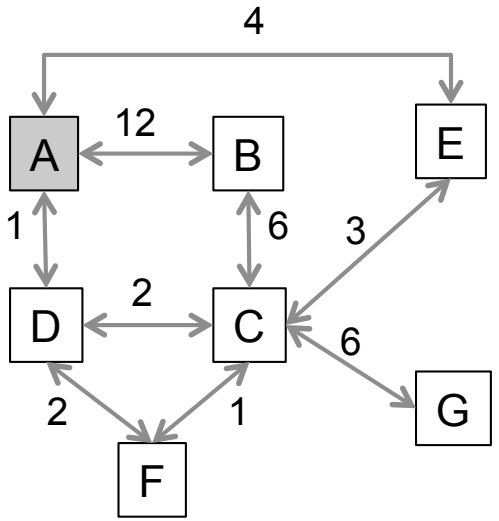
G	
C	6

2. Chaque nœud reconstruit toute la topologie en combinant tous les paquets LSP

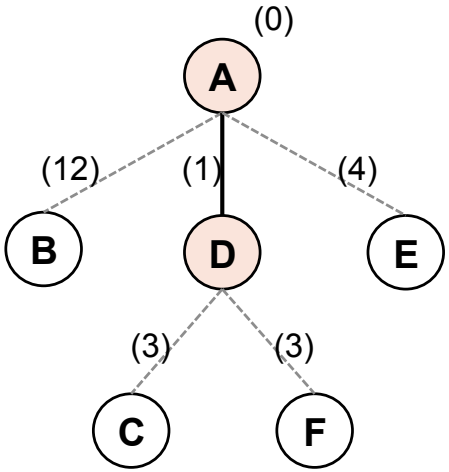
EXEMPLE ROUTAGE PAR ÉTAT DE LIEN

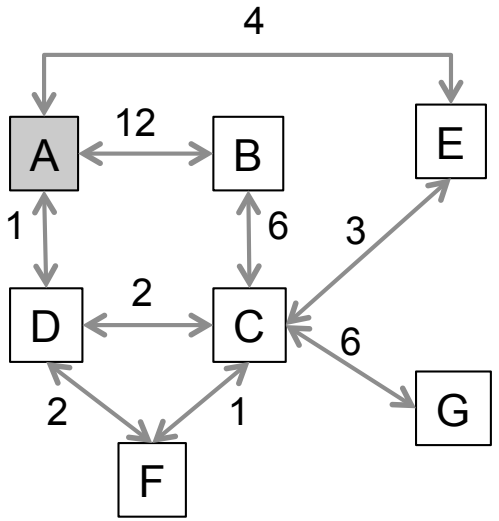
3. Chaque nœud exécute Dijkstra en considérant qu'il est le sommet source



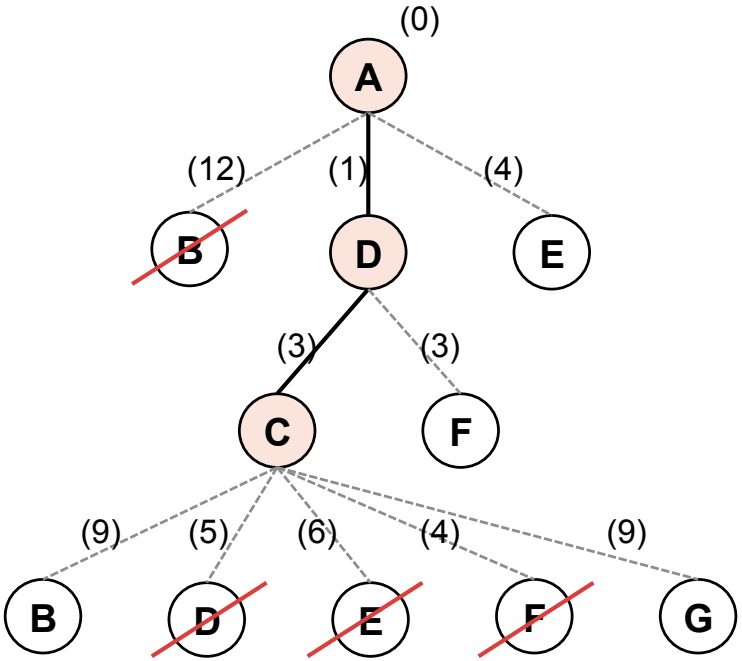


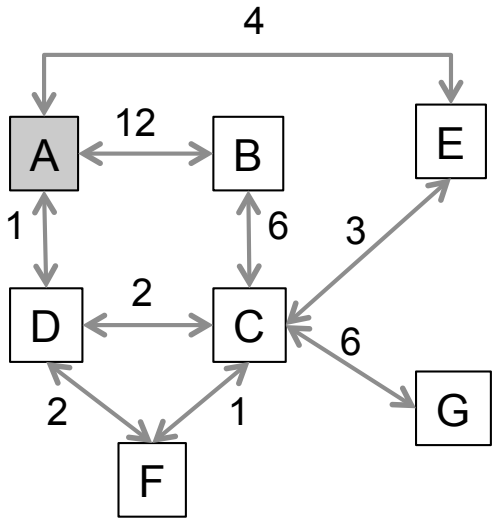
Exemple routage par état de lien



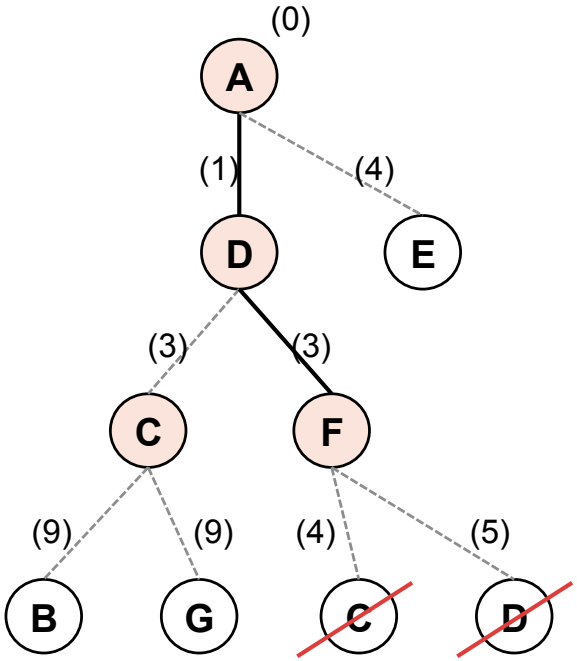


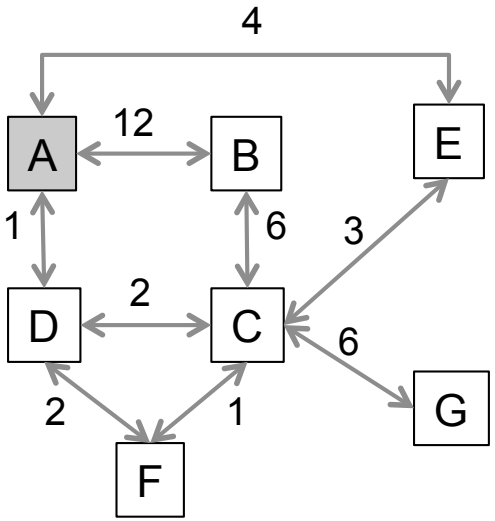
EXEMPLE ROUTAGE PAR ÉTAT DE LIEN



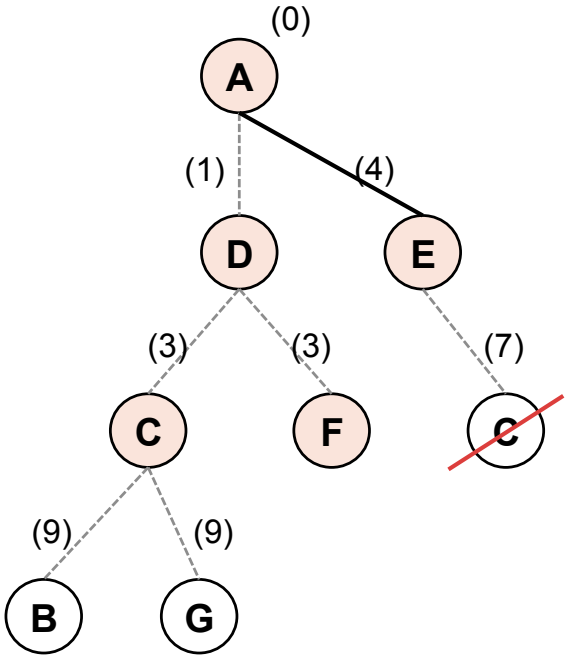


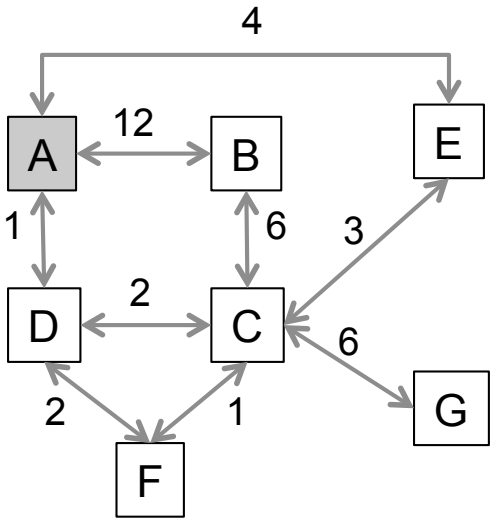
Exemple routage par état de lien



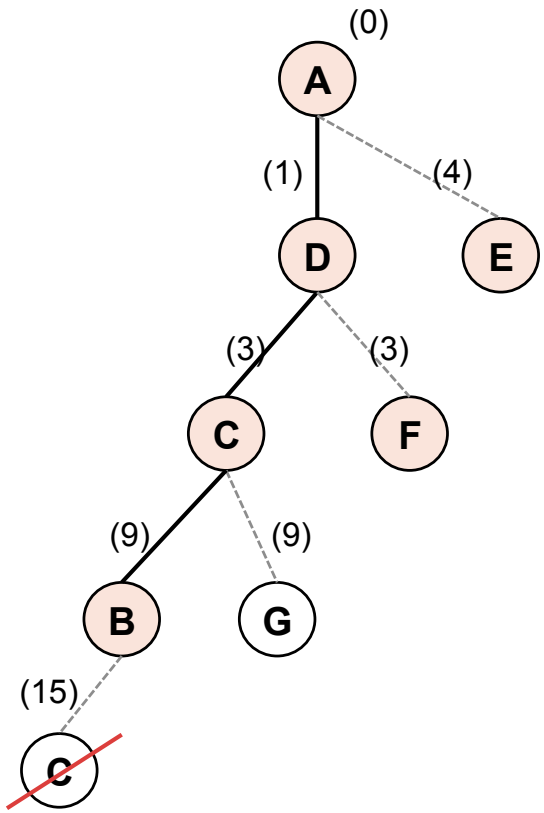


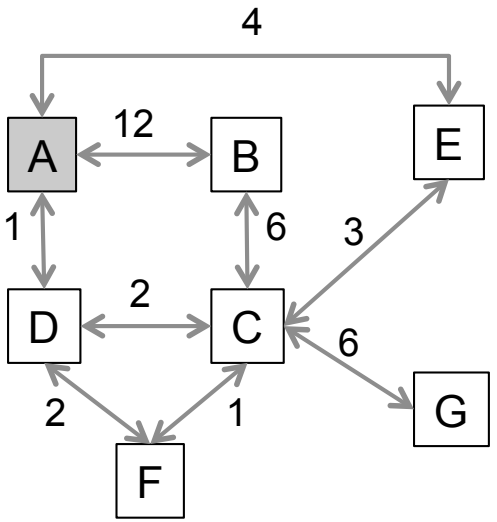
Exemple routage par état de lien



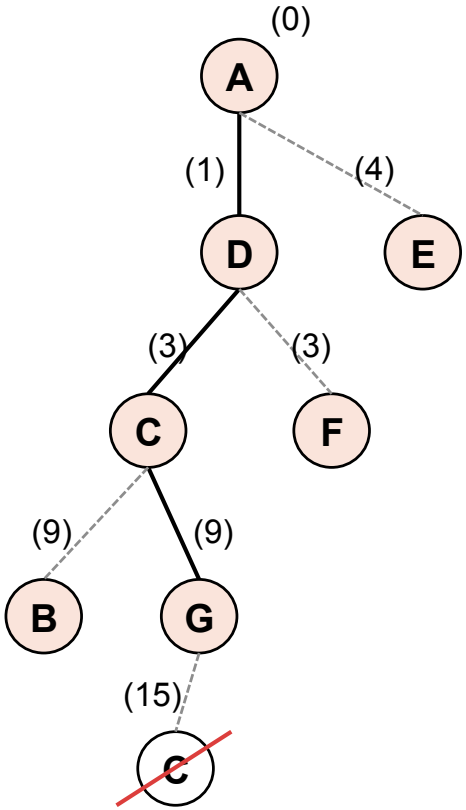


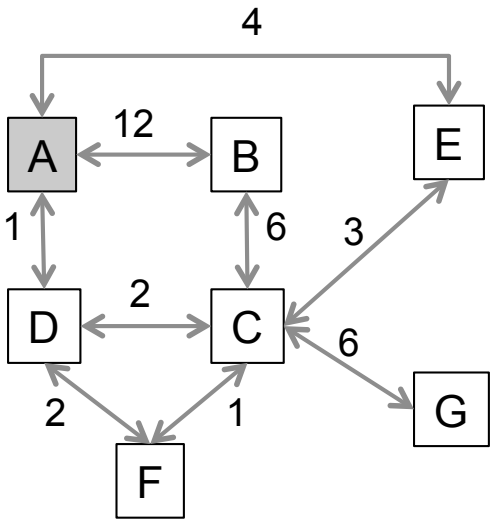
EXEMPLE ROUTAGE PAR ÉTAT DE LIEN



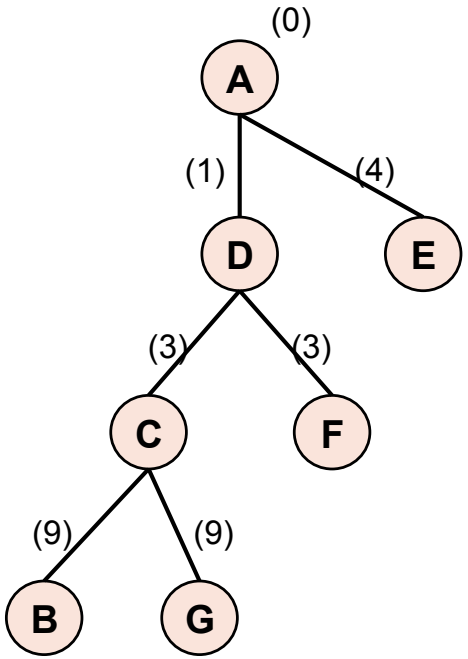


EXEMPLE ROUTAGE PAR ÉTAT DE LIEN





Exemple routage par état de lien



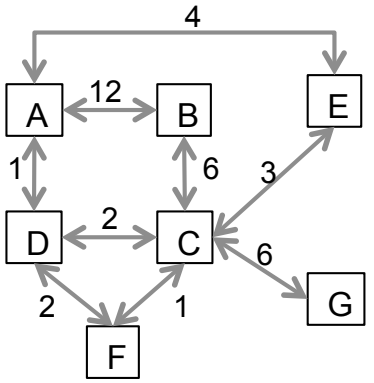


Table de routage de A

Destinat.	Prochain saut	Coût
A	local	0
B	D	9
C	D	3
D	D	1
E	E	4
F	D	3
G	D	9

GESTION DE LA DYNAMIQUE

Ajout d'un lien ou nœud

- Ajout et diffusion du paquet LSP du nouveau nœud
- Les LSP anciens sont mis à jour avec le nouveau lien

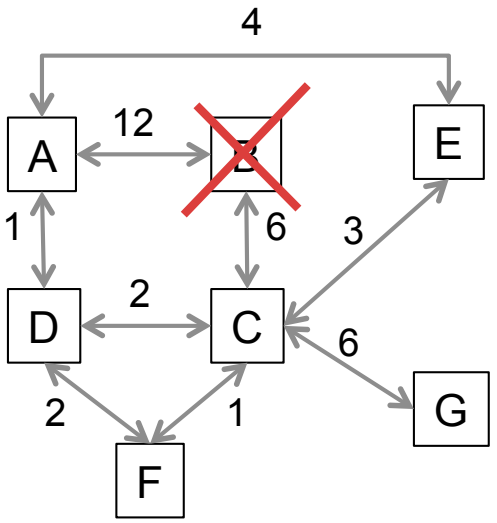
Panne d'un lien ou nœud

- Les nœuds adjacents sont informés de la panne et modifient leur paquet d'état de lien (LSP)
- Diffusion des nouveaux LSP et routes recalculées

LSP

A	
B	∞
D	1
E	4

C	
B	∞
D	2
E	3
F	1
G	6



Vecteur distance vs. Etat de lien

Critère	Vecteur distance	Etat de lien
Algorithme	Bellman-Ford	Dijkstra
Choix du chemin	Plus court via fonction de coût	Plus court via fonction de coût
Vue topologie	Vue des voisins et de la distance des autres nœuds	Vue globale de toute la topologie
Rapidité de convergence	Lent	Plus rapide
Bande passante	Faible (tout le vecteur aux voisins seulement)	Faible (l'état des voisins mais à tout le monde)
Calcul / Mémoire	Faible	Modeste

Routage hiérarchique

Le routage est fait par « région »

- Chaque routeur d'un sous-réseau connaît les informations pour router dans sa région, mais ne connaît rien quant à la structure interne d'une autre région.
- Il connaît éventuellement la route pour atteindre une autre région. (au moins une par région)
- Plusieurs niveaux de hiérarchie peuvent être mis en place (en fonction de la taille du réseau).

ROUTAGE HIÉRARCHIQUE

Exemple

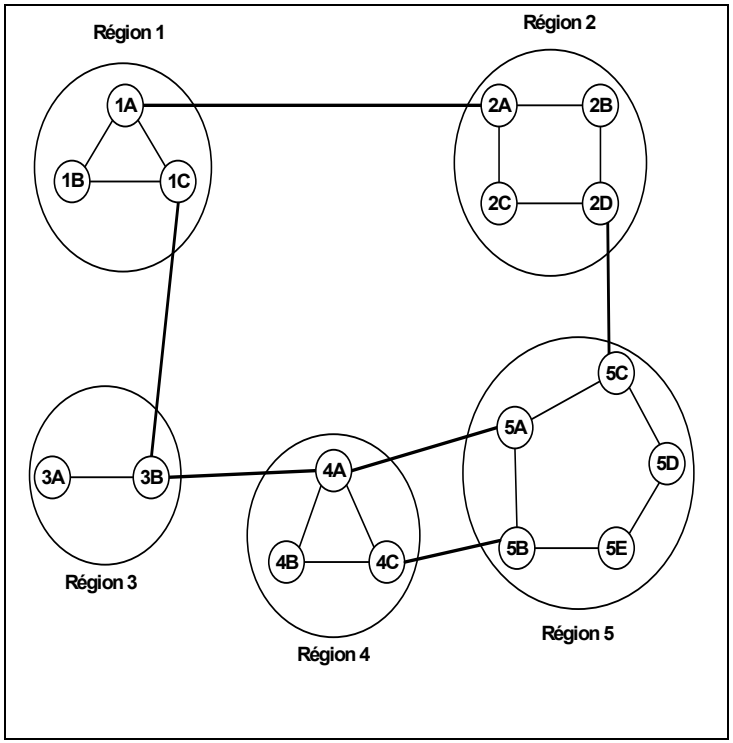


Table de routage 1A

Destination	Prochain saut	Coût
1B	1B	1
1C	1C	1
Région 2	2A	1
Région 3	1C	2
Région 4	1C	3
Région 5	1C	4

PROTOCOLES DE ROUTAGE : STANDARDS TCP/IP

		Protocoles dérivés de VECTEUR DISTANCE	Protocoles dérivés de ETAT de LIAISON
TCP/IP	*routage intra-domaine*	**RIP** (Routing Information Protocol)	**OSPF** (Open Short Path First)
	routage inter-domaine	**EGP** (External Gateway Protocol) **BGP (vecteur chemin)** (Border Gateway Protocol)	