

# Exercices Dirigés

Unité d'enseignement RSX101

## Réseaux et protocoles pour l'Internet -Routage IP Généralités-

2020-2021

*Ce support a été élaboré par l'équipe enseignante "Réseaux et protocoles", auteurs principaux M. Gressier Soudan et Gérard Florin (ancien professeur responsable de l'ue de réseaux et protocoles, à la retraite).*



# ED7•Couche Réseau :

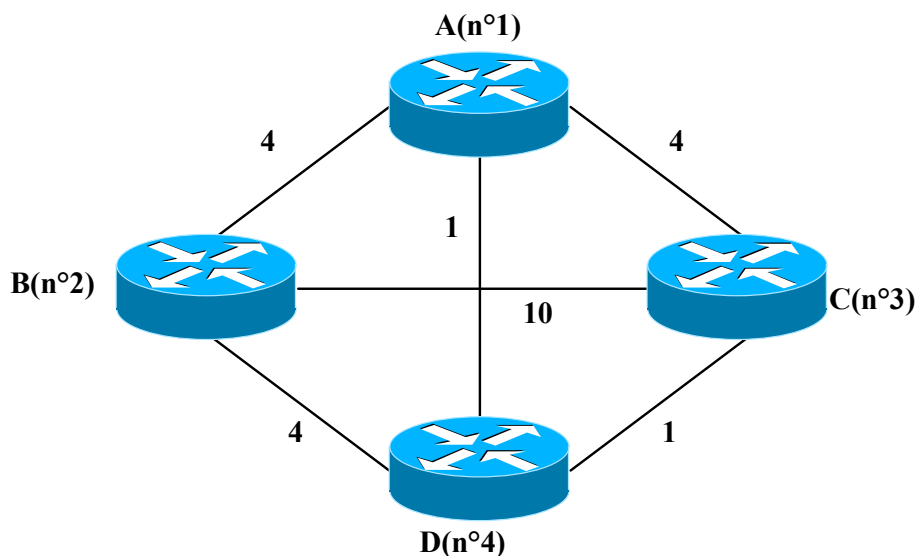
## Routage IP

### Exercice 1 : Routage à vecteur de distance, un exemple RIP (pour information)

RIP (Routing Information Protocol) dans l'Internet n'est plus utilisé quasiment. Toutefois, c'est un bon exemple de routage par vecteur de distance, et cette approche est utilisée dans les réseaux wifi ad-hoc. Pédagogiquement cet exercice conserve toujours un intérêt. RIP s'inspire des travaux sur les algorithmes à vecteur de distance de Bellman-Ford mais aussi parfois référencé comme travaux de Ford-Fulkerson.

Si on doit considérer RIP d'un point de vue opérationnel, il faut alors utiliser RIPv2 (RFC 2453). Pour simplifier le discours, nous utiliserons dans l'exercice le terme RIP, sous entendu RIPv2. RIPv1 possède plusieurs défauts rédhibitoires : il est classfull (l'Internet a basculé en classless depuis l'avènement de CIDR), il n'a qu'une seule métrique, il calcule les chemins optimaux avec le nombre de sauts, donc il ne prend pas en compte les indications du champ TOS. Enfin, il souffre du comptage à l'infini

Le graphe ci-dessous représente un réseau informatique : les sommets : A, B, C, D représentent les noeuds de commutation et les arcs les liaisons bidirectionnelles. La valeur portée sur les arcs représente une distance de communication point à point qui est supposée être identique dans les deux sens. Cette distance matérialise un coût de communication qui sert dans la suite de l'ED pour déterminer un meilleur chemin et fabriquer les tables de routage des différents noeuds.



Plus loin dans l'exercice, on utilise des numéros pour désigner les noeuds de commutation : A est le n° 1, B est le n° 2, C est le n° 3, et D est le n° 4. C'est plus facile car on utilise des matrices pour les calculs.

### Question 1 Calcul du routage centralisé

Pour calculer le routage optimal on utilise l'algorithme centralisé suivant (qui n'est pas le meilleur algorithme centralisé de calcul de chemin minimal dans un graphe). Calculer à l'aide de cet algorithme le routage optimal du réseau. Déterminer, en utilisant la matrice P, l'arbre de routage (sink tree) et la table de routage du noeud A.

#### CONSTANTES

```
N: entier ; % Nombre de noeuds;
Type Nom_de_noeud: entier dans (1..N); % nom des noeuds de commutation;
C : tableau (N,N) de réels ;

% cout de transmission en point à point;

C(i,i)=0 pour tout i et C(i,j)=+infini si il n'y a pas de liaison entre i et j;

Quand il y en a une, le coût a été porté sur l'arc du graphe ci-dessus %;
```

#### VARIABLES

```
V: tableau (N,N) de réels ;

% A la fin de l'algorithme V(i,j) est égal au coût du chemin de valeur minimale de i
vers j %;

P: tableau (N,N) de Nom_de_noeuds;

% A la fin de l'algorithme P(i,j) est le successeur de i dans le chemin optimal de i
vers j%;

Iter: entier; % numéro d'itération%;

cout: réel;

i,j,k,r: Nom_de_noeuds;
```

#### CORPS DE PROCEDURE (Ford)

```
% Initialisation ;

pour i=1 a N
  pour j=1 a N
    V(i,j) := C(i,j);
    Si C(i,j) différent de +infini faire P(i,j):=j;
  finsi
finpour
finpour
```

```

% Itérer N-1 fois %;
Pour iter=1 a N-1;
  % Pour chaque noeud origine d'un chemin %;
  Pour i=1 a N
    % phase A de l'algorithme : %
    % Pour chaque noeud extrémité d'un chemin %;
    Pour j=1 a N
      cout:=+ infini;
      % Chercher parmi tous les voisins de i celui par lequel
      passe le meilleur chemin vers j %;
      Pour k=1 a N
        Si C(i,k) différent de +infini° faire
          Si C(i,k)+ V(k,j) inférieure cout faire
            cout:=C(i,k)+V(k,j);
            r:=k;
          finsi
        finsi
      fin pour
      %Si la valeur trouvée est meilleure que celle contenue
      dans V noter le nouveau chemin %;
      Si V(i,j)> cout faire
        V(i,j):=cout;
        P(i,j):=r;
      finsi
    finpour
  finpour
finpour

```

### Correction :

Du graphe on déduit la matrice de coût C, :

C	1-A	2-B	3-C	4-D
1-A	0	4	4	1
2-B	4	0	10	4
3-C	4	10	0	1
4-D	1	4	1	0

On associe numéro et lettre pour mieux identifier les routeurs du dessin dans la matrice ci-dessus. Cette correspondance est toujours valable pendant tout l'exercice.

Comme les liens ont le même coût quel que soit leur sens, on constate une symétrie diagonale dans la matrice C.

L'initialisation des matrices P et V donne :

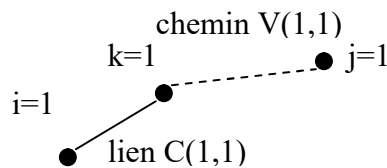
V	1	2	3	4
1	0	4	4	1
2	4	0	10	4
3	4	10	0	1
4	1	4	1	0

P	1	2	3	4
1	1	2	3	4
2	1	2	3	4
3	1	2	3	4
4	1	2	3	4

On bénéficie de propriété de symétrie. Dans  $V$  les valeurs calculées sont identiques de part et d'autre de la diagonale, car les liens ne sont pas orientés (coût identique dans les deux sens). Si les coûts étaient différents d'un sens à l'autre, si on considérait des arcs au lieu de liens, il n'y aurait pas cette symétrie. Certains liens sont disymétriques par nature, les liens ADSL par exemples, le débit descendant est différent du débit montant souvent largement inférieur.

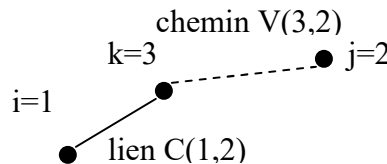
On commence maintenant la phase de calcul du chemin de coût minimum :

- iter :=1
  - i:=1
    - j:=1, cout := +infini
      - k :=1 cout :=0, r := 1



le cas  $i=k=j=1$  ne fait pas sens, je vous l'accorde mais c'est l'algorithme... dans l'absolu on ne travaille pas sur les chemins de la diagonale de  $V$ . En effet les autres itérations qui étudient cout + chemin en partant de 1 pour aller à 1 ne font pas plus de sens.

- k = 2, 3, 4 ne font pas diminuer cout
- $V(1,1)$  n'est pas supérieur à cout,  $V(1,1)$  ne change pas, et  $P(1,1)$  non plus
- j := 2, cout := + infini
  - k:= 1 cout := 4, r:= 1
  - k= 2, 3, 4 ne vont pas diminuer cout, le dessin ci-dessous illustre  $i=1$ ,  $J=2$ , et  $k=3$



- $V(1,2)$  n'est pas supérieur à cout,  $V(1,2)$  ne change pas, et  $P(1,2)$  non plus
- j := 3, cout := + infini
  - k:= 1 cout := 4, r:= 1
  - k =2, 3 ne changent rien
  - k := 4 cout := 2, r:= 4
  - $V(1,3)$  est supérieur à cout, donc on peut changer  $V(1,3)$  vaut 2 maintenant, et  $P(1,3)$  vaut 4
- j:=4, cout := +infini
  - k :=1 cout :=1, r := 1
  - k = 2, 3, 4 ne font pas diminuer cout
  - $V(1,4)$  n'est pas supérieur à cout,  $V(1,4)$  ne change pas, et  $P(1,4)$  non plus

Avec la propriété de symétrie, appliquée à V et P, les matrices sont les suivantes :

V	1	2	3	4
1	0	4	2	1
2	4	0	10	4
3	2	10	0	1
4	1	4	1	0

P	1	2	3	4
1	1	2	4	4
2	1	2	3	4
3	4	2	3	4
4	1	2	3	4

Quelques remarques pour diminuer le nombre de calculs :

- La matrice V est symétrique
- Quand  $i = j$  le cout est nul, pas la peine de faire le calcul

Il ne nous reste donc que 3 calculs pour obtenir V sur la boucle  $\text{iter} = 1$

○  $i := 2$

- $j := 1$  pas la peine, on l'a obtenu par la symétrie de la matrice V
- $j = 2$  on ne fera pas mieux que 0 puis qu'on va de 2 à 2
- $j = 3$  cout := +infini
  - $k := 1$  cout := 8,  $r := 1$
  - $k = 2$ , 3 ne font pas diminuer cout
  - $k := 4$  cout := 5,  $r := 4$
  - $V(2,3)$  est supérieur à cout, donc on peut changer  $V(2,3)$  vaut 5 maintenant, et  $P(2,3)$  vaut 4
  - On a du coup deux nouvelles matrices (en tenant compte de la symétrie du problème) :

V	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	2	5	0	1
4	1	4	1	0

P	1	2	3	4
1	1	2	4	4
2	1	2	4	4
3	4	4	3	4
4	1	2	3	4

- $j = 4$  cout := +infini
  - $k := 1$  cout := 5,  $r := 1$
  - $k := 2$  cout := 4,  $r := 2$
  - $k = 3, 4$  pas d'amélioration de cout
  - $V(2,4)$  n'est pas supérieur à 4, donc  $V(2,4)$  et  $P(2,4)$  ne changent pas

- $i:=3$ 
  - $j:=1$  et  $j:=2$  pas la peine, on les a obtenus par la symétrie de la matrice V
  - $j = 3$  on ne fera pas mieux que 0 puis qu'on va de 3 à 3
  - $j = 4$  cout := +infini
    - $k:=1$  cout :=5,  $r:=1$
    - $k = 2$  pas d'amélioration
    - $k:=3$  cout :=1,  $r:=3$
    - $k = 4$  pas d'amélioration
    - $V(3,4)$  n'est pas supérieur à 1, donc  $V(3,4)$  et  $P(3,4)$  ne changent pas

Finalement le résultat pour iter = 1 est :

V	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	2	5	0	1
4	1	4	1	0

P	1	2	3	4
1	1	2	4	4
2	1	2	4	4
3	4	4	3	4
4	1	2	3	4

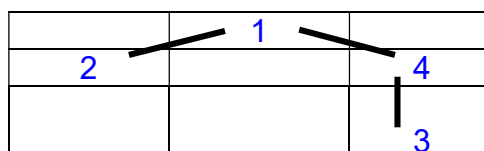
Pour iter = 2 rien ne change, idem pour iter = 3, donc les matrices V et P ci-dessus sont les bonnes.

Du coup, la table de routage du nœud A (n°1) s'extraite de V et de P :

A(n°1)	1	2	3	4
Next Hop P(1,.)	1	2	4	4
Coût V(1,.)	0	4	2	1

Habituellement, on présente la table de routage sous la forme d'une suite de colonne.

Arbre de routage à partir du nœud 1 :



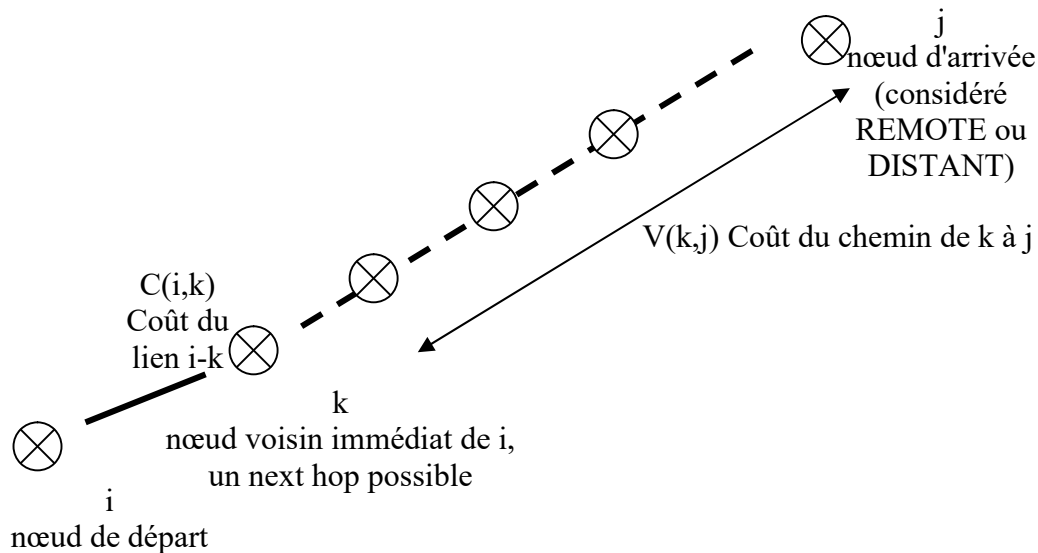
### Remarques :

1. Le calcul des tables de routage se fait sur un seul calculateur. Chaque table de routage est envoyée au routeur correspondant. Comme cet envoi est asynchrone, et que les tables arrivent dans des délais variés fonction de la

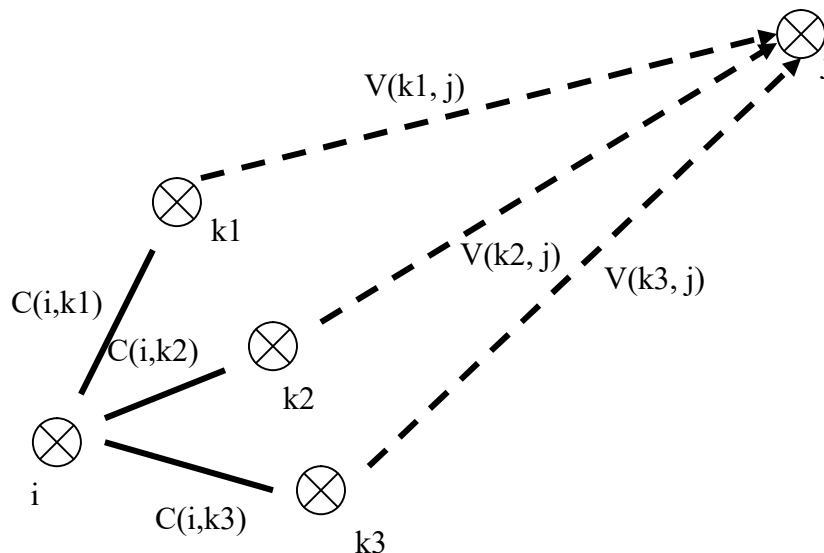
distance du routeur par rapport au site de calcul, tous les routeurs ne sont pas mis à jour en même temps, d'où une certaine instabilité potentielle du routage global.

Le routage dans Internet est plutôt un calcul décentralisé. RIP reprend le schéma de l'algorithme qu'on vient de voir cf question suivante.

2. Tous les éléments pour le calcul du routage tiennent sur le rôle des indices  $i$ ,  $j$ ,  $k$  :



Comme un nœud à plusieurs voisins immédiats, on cherche le meilleur voisin  $k$  parmi tous les voisins de  $i$  pour aller à  $j$ . Ce qui donne avec plusieurs voisins  $k_1$ ,  $k_2$ ,  $k_3$  :



### Question 2 : Algorithme réparti adaptatif (Belman-Ford/Mac Quillan)

Dans une version d'Arpanet, le routage adaptatif était réalisé en utilisant un algorithme adaptatif réparti dérivé du précédent. Dans cet algorithme le nœud  $i$  calcule uniquement la  $i$ ème ligne des matrices  $V$  et  $P$  (soit le coût de transmission d'un message de  $i$  vers tous les autres nœuds du réseau et l'adjacent préféré de  $i$  vers tous les nœuds du réseau). Pour cela, il envoie périodiquement **à ses voisins**



$V(i, \cdot)$  et reçoit de chacun d'entre eux la ligne de  $V$  correspondante. L'algorithme s'adapte aux variations des coûts qui sont mesurés à chaque itération.

```

Algorithme Arpa(i)
cycle
    Attendre T secondes;
    pour tout voisin k de i % voisin au sens interface réseau
        Envoyer  $V(i, \cdot)$ ;
        Recevoir  $V(k, \cdot)$ ;
        Mesurer  $C(i, k)$ ;
    finpour
    -- Les données sont prêtes pour une itération du calcul
    Appliquer la partie A de l'algorithme précédent pour
trouver
    les nouvelles valeurs de  $V(i, \cdot)$  et  $P(i, \cdot)$ ;
    Déterminer la nouvelle table de routage de i;
fincycle

```

La mesure avec l'opération `Mesurer  $C(i, k)$`  peut s'effectuer de plusieurs façons différentes. Par exemple on peut considérer que le coût d'une ligne est infini si sur une interface, la file de messages en émission vers un destinataire est pleine, et vide en réception depuis celui-ci.

On suppose qu'à l'instant  $t=0$  tous les noeuds ont calculé les valeurs respectives de  $V(i, \cdot)$  et  $P(i, \cdot)$  et la table de routage correspondante. La liaison (3,4) tombe en panne. Chaque noeud commence une étape du cycle de calcul.

Pour réaliser la phase A de l'algorithme le noeud  $i$  calcule, à la  $n$ -ème étape du calcul, une matrice  $S_i^{(n)}$  où  $S_i^{(n)}(k, j)$  est l'évaluation par  $i$  du coût pour aller à  $j$  en passant par  $k$ .

Quelle relation lie  $S_i^{(n)}(k, j)$ ,  $V_k^{(n-1)}(k, j)$  et  $C^{(n)}(i, k)$  (valeur du chemin optimal vers  $j$  calculés par  $k$  au cycle précédent et coût mesuré par  $i$  au cycle courant) ?

En déduire la relation liant  $S_i^{(n)}$  et  $V^{(n)}(i, \cdot)$

Appliquer l'algorithme pour la première itération (numérotée  $n$ ).

Il est recommandé de retracer le graphe à grande échelle et d'associer au noeud  $i$  la matrice  $S_i$ .

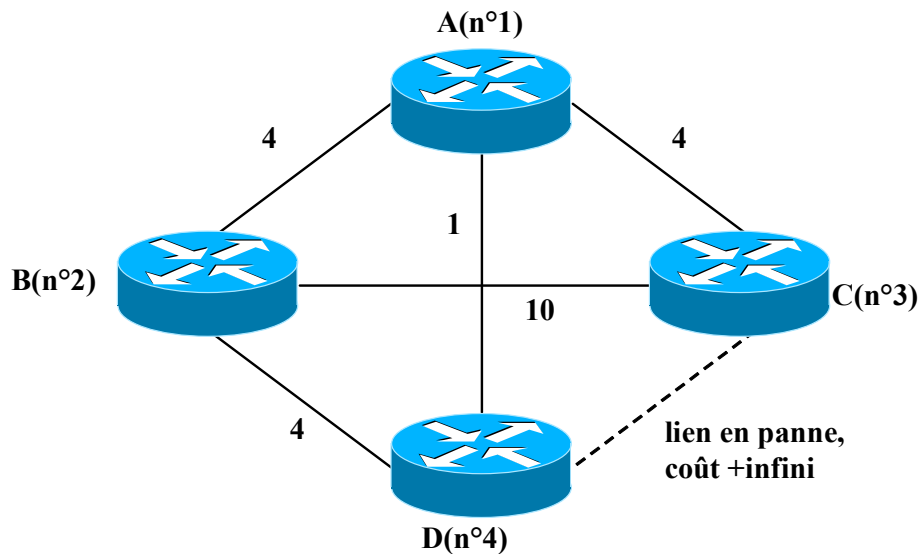
### Correction :

Si  $S_i^{(n)}(k, j)$  est l'évaluation par  $i$  du coût pour aller à  $j$  en passant par  $k$  à l'étape de calcul  $n$ , alors  $S_i^{(n)}(k, j) = C^{(n)}(i, k) + V_k^{(n-1)}(k, j)$ , cette relation s'obtient facilement d'après l'observation faite dans la figure élaborée ci-dessus.

On en déduit que  $V_i^{(n)}(i, j) = \min_k S_i^{(n)}(k, j)$  où  $k$  sont les voisins immédiats de  $i$

Au temps  $t_0$ , la liaison D-C tombe en panne, son coût devient +infini, seuls D et C sont capables de mesurer ce coût, les autres routeurs vont l'apprendre grâce à l'échanges des tables de routage et de l'exécution du calcul du routage de coût minimum de Mac Quillan.

Le principe sous-jacent de RIP est un fonctionnement de type rumeur, ou encore "l'homme qui a vu l'homme qui a vu l'homme qui a vu l'ours".



Au temps  $t_0$  les différents sites lancent une phase du calcul du routage optimal. Dans cette question le calcul qu'effectue chaque routeur est local, il ne faut pas l'oublier. Donc pour tout nœud  $i$  en considérant qu'on passe de l'étape  $n-1$  à l'étape  $n$  :

```

pour tout voisin  $k$  de  $i$ 
    Envoyer  $V^{(n-1)}(i, \bullet)$ ; %envoyer sa table de routage
    Recevoir  $V^{(n-1)}(k, \bullet)$ ; %recevoir la table de routage de tous ses voisins
    Mesurer  $C^{(n)}(i, k)$ ;
finpour
  
```

**ATTENTION** : LE CORRIGE PRESENTE CE QU'IL SE PASSE LINEAIREMENT (NŒUD APRES NŒUD) MAIS EN REALITE TOUT SE PASSE EN PARALLELE ET DE FACON ASYNCHRONE SUR CHAQUE NŒUD. CE N'EST PAS SIMPLE A IMAGINER MAIS C'EST-CE QU'IL FAUT S'EFFORCER DE FAIRE POUR BIEN SAISIR L'EXECUTION DE RIP ET TOUT LE PARALLELISME SOUS-JACENT.

**Remarque** : On peut aussi se poser la question "Comment C connaît la table de routage de D, et, comment D connaît la table de routage de C ?" puisque le lien direct entre C et D est coupé. On verra plus loin cet aspect sous différents angles.

### Une première solution plutôt formelle.

- Pour le nœud A (n°1) voila les étapes :

Il connaît par calcul à l'étape précédente :

A(n°1)	1	2	3	4
$P^{(n-1)}(1, \bullet)$	1	2	4	4
$V^{(n-1)}(1, \bullet)$	0	4	2	1

Il reçoit de ses voisins immédiats (dédit de la réponse à la question 1) :

B(n°2)	1	2	3	4
$P^{(n-1)}(2, \bullet)$	1	2	4	4
$V^{(n-1)}(2, \bullet)$	4	0	5	4

$C^{(n-3)}$	1	2	3	4
$P^{(n-1)}(3,.)$	4	4	3	4
$V^{(n-1)}(3,.)$	2	5	0	1

On peut se demander comment les informations de routage parviennent à A car d'après sa table, C devrait envoyer à D pour atteindre A alors que le lien est coupé manifestement. En fait, c'est simple, les tables RIP sont envoyées sur toutes les interfaces d'un routeur par un multicast IP à l'adresse 224.0.0.9 (RIP V2) et ne dépassent pas ce routeur voisin à cause d'un TTL à 1.

$D^{(n-4)}$	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

On obtient alors pour résoudre le calcul du routage de A la matrice  $C^{(n)}(1,.)$  mesurée et la matrice  $V_1^{(n)}$

$C^{(n)}(1,.)$	1	2	3	4
1	0	4	4	1

$V_1^{(n-1)}$	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	2	5	0	1
4 <sup>1</sup>	1	4	1	0

Pour faciliter le calcul, et j'espère la compréhension, on va représenter C, V et S dans un même tableau, C sera mis en colonne :

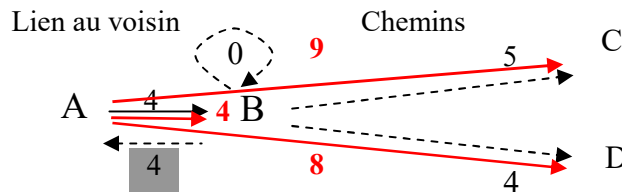
i	voisin		destination				$S_1^{(n)} j \rightarrow$				
	$C^{(n)}(1,.)$	k pour $V_1^{(n-1)}$	1	2	3	4		1	2	3	4
1	0	1	0	4	2	1	/				
1	4	2	4	0	5	4	2		4	9	8
1	4	3	2	5	0	1	3		9	4	5
1	1	4	1	4	1	0	4		5	2	1

- Première colonne : nœud de départ, ici 1 (A)
- Deuxième colonne : coût du lien pour aller de 1 à k (1-A, 2-B, 3-C, 4-D) k étant un voisin immédiat de 1 (A)
- Troisième colonne : k, le voisin par lequel on passe
- Les 4 autres colonnes correspondant à la matrice des coûts de chemin V construite au nœud 1 en recevant les tables de routage des autres routeurs de l'itération de calcul précédente, pour nous n-1.
- En grisé les valeurs sans intérêt :
  - départ et arrivée  $i = j = 1$ ,
  - où on est soi-même la destination (colonne associée à la destination  $j=1$ ), partir de 1 pour aller à 1 quel que soit le voisin sera plus coûteux que 0.

<sup>1</sup> Il serait intéressant de refaire le calcul en faisant l'hypothèse que la table de routage de C n'est jamais reçue par A. On remplacerait par une valeur par défaut, +infini par exemple.

- où on est soi-même le successeur (ligne associée à  $k=1$  comme voisin), ça perd son sens car on a un coût de départ du premier lien qui vaut 0, et la valeur du chemin de  $k$  à  $j$  est donné par les autres lignes.
- Les colonnes qui suivent correspondent à  $S$  au nœud 1.

On peut représenter cet ensemble de calcul à l'aide d'un schéma. On prend par exemple  $k = 2$  comme voisin. On se rappelle qu'on part de A, et qu'on cherche à atteindre les autres nœuds du réseau.



Pour chaque colonne on prend le voisin  $k$  (next hop) tel que le coût pour aller à la destination est minimum. On en déduit  $V^{(n)}(1, .)$  :

A(n°1)	1	2	3	4
$V^{(n)}(1,.)$	0	4	2	1

La matrice des successeurs correspondante est obtenue en sélectionnant le  $k$  qui permet d'obtenir le chemin à coût minimum pour la destination cible considérée :

A(n°1)	1	2	3	4
$P^{(n)}(1,.)$	1	2	4	4

On constate que pour le nœud A (n°1) rien n'a changé par rapport à l'étape précédente (n-1). C'est normal, les tables qu'il a reçues ne tiennent pas compte pour l'instant de la panne qui n'est en fait détectée que pendant cette phase. On verra que pour D et C cela va changer dès cette étape.

#### • Pour le nœud B (n°2) voila les étapes :

Il connaît :

B(n°2)	1	2	3	4
$P^{(n-1)}(2,.)$	1	2	4	4
$V^{(n-1)}(2,.)$	4	0	5	4

Il reçoit des autres nœuds :

A(n°1)	1	2	3	4
$P^{(n-1)}(1,.)$	1	2	4	4
$V^{(n-1)}(1,.)$	0	4	2	1

C(n°3)	1	2	3	4
$P^{(n-1)}(3,.)$	4	4	3	4
$V^{(n-1)}(3,.)$	2	5	0	1

D(n°4)	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

On obtient alors pour résoudre le calcul : la matrice  $C^{(n)}(2,.)$  mesurée, et, la matrice  $V_2^{(n-1)}$  construite à l'aide des vecteurs de distance reçu des voisins immédiats.

$C^{(n)}(2,.)$	1	2	3	4
2	4	0	10	4

$V_2^{(n-1)}$	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	2	5	0	1
4	1	4	1	0

En fait  $V_i^{(n-1)}$  sera la même pour tous les nœuds compte tenu du fait que le réseau qui nous sert d'exemple est complètement maillé. On procède alors comme avec le nœud A.

i	$C^{(n)}(2,.)$	k pour $V_2^{(n-1)}$	1	2	3	4	$S_1^{(n)} j \rightarrow$	1	2	3	4
2	4	1	0	4	2	1	1	4	/	6	5
2	0	2	4	0	5	4	2	/	/	/	/
2	10	3	2	5	0	1	3	12	/	10	11
2	4	4	1	4	1	0	4	5	/	5	4

On en déduit  $V^{(n)}(2,.)$  :

$B(n^{\circ}2)$	1	2	3	4
$V^{(n)}(2,.)$	4	0	5	4

La matrice des successeurs correspondante est obtenue en sélectionnant le k qui permet d'obtenir le chemin à coût minimum pour la destination cible considérée :

$B(n^{\circ}2)$	1	2	3	4
$P^{(n)}(2,.)$	1	2	4	4

On constate que pour le nœud B ( $n^{\circ}2$ ) cela ne change pas par rapport à l'étape précédente ( $n-1$ ).

- Pour le nœud C ( $n^{\circ}3$ ) voila les étapes :

Il connaît :

$C(n^{\circ}3)$	1	2	3	4
$P^{(n-1)}(3,.)$	4	4	3	4
$V^{(n-1)}(3,.)$	2	5	0	1

Il reçoit (dédit de la réponse à la question 1) :

$A(n^{\circ}1)$	1	2	3	4
$P^{(n-1)}(1,.)$	1	2	4	4
$V^{(n-1)}(1,.)$	0	4	2	1

$B(n^{\circ}2)$	1	2	3	4
$P^{(n-1)}(2,.)$	1	2	4	4
$V^{(n-1)}(2,.)$	4	0	5	4

$D(n^{\circ}4)$	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

Pour D, c'est mis en rose pour figurer la difficulté à recevoir cette table sachant que le lien C-D est coupé.

On obtient alors pour résoudre le calcul la matrice  $C^{(n)}(3,.)$  mesurée et la matrice  $V_3^{(n-1)}$

$C^{(n)}(3,.)$	1	2	3	4
3	4	10	0	+infini

$V_3^{(n-1)}$	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	2	5	0	1
4	1	4	1	0

On répète cette matrice à chaque fois, bien que le lecteur aura compris mais comme ça l'exercice est fait en littéral.

On procède alors comme avec le nœud A.

i	$C^{(n)}(3,.)$	k pour $V_3^{(n-1)}$	1	2	3	4	$S_1^{(n)} j \rightarrow$	1	2	3	4
3	4	1	0	4	2	1	1	4	8	/	5
3	10	2	4	0	5	4	2	14	10	/	14
3	0	3	2	5	0	1	3	/	/	/	/
3	+infini	4	1	4	1	0	4	+infini	+infini	/	+infini

Avec le calcul on remarque que pour C (n°4) recevoir ou non la table de routage de D n'a pas d'importance puisque le coût +infini va "absorber" toute valeur du coût du chemin dans cette table.

On en déduit  $V^{(n)}(3,.)$  :

$C(n^{\circ}3)$	1	2	3	4
$V^{(n)}(3,.)$	4	8	0	5

La matrice des successeurs correspondante est obtenue en sélectionnant le k qui permet d'obtenir le chemin à coût minimum pour la destination cible considérée :

$C(n^{\circ}3)$	1	2	3	4
$P^{(n)}(3,.)$	1	1	3	1

On constate que pour le nœud C (n°3) cela change par rapport à l'étape précédente (n-1). En effet, C tient immédiatement compte dans son calcul de la perte du lien D-C.

• **Pour le nœud D (n°4) voila les étapes :**

Il connaît sa table :

$D(n^{\circ}4)$	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

Il obtient de ses voisins :

$A(n^{\circ}1)$	1	2	3	4
$P^{(n-1)}(1,.)$	1	2	4	4
$V^{(n-1)}(1,.)$	0	4	2	1

<b>B(n°2)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (2,,)</b>	1	2	4	4
<b>V<sup>(n-1)</sup> (2,,)</b>	4	0	5	4

<b>C(n°3)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (3,,)</b>	4	4	3	4
<b>V<sup>(n-1)</sup> (3,,)</b>	2	5	0	1

On obtient alors pour résoudre le calcul la matrice  $C^{(n)}(4,.)$  mesurée et  $V_4^{(n-1)}$

<b>C<sup>(n)</sup> (4,,)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>4</b>	1	4	+infini	0

<b>V<sub>4</sub><sup>(n-1)</sup></b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	4	2	1
<b>2</b>	4	0	5	4
<b>3</b>	2	5	0	1
<b>4</b>	1	4	1	0

i	C <sup>(n)</sup> (4,,)	k pour V <sub>3</sub> <sup>(n-1)</sup>	1	2	3	4	S <sub>1</sub> <sup>(n)</sup> j->	1	2	3	4
4	1	1	0	4	2	1	1	1	5	3	/
4	4	2	4	0	5	4	2	8	4	9	/
4	+infini	3	2	5	0	1	3	+infini	+infini	+infini	/
4	0	4	1	4	1	0	4	/	/	/	/

On en déduit  $V^{(n)}(4,.)$  :

<b>D(n°4)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>V<sup>(n)</sup> (4,,)</b>	1	4	3	0

La matrice des successeurs correspondante est obtenue en sélectionnant le k qui permet d'obtenir le chemin à coût minimum pour la destination cible considérée :

<b>D(n°4)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n)</sup> (4,,)</b>	1	2	1	4

On constate que pour le nœud D (n°4) aussi cela change par rapport à l'étape précédente (n-1). En effet, D tient immédiatement compte dans son calcul du routage à coût minimum de la perte du lien D-C.

### Bilan de la question :

- A et B dans leur calcul du routage à coût minimum n'ont pas pris en compte la panne parce qu'ils en sont éloignés et qu'on ne les a pas informés du changement de topologie. En effet, les nœuds apprennent la nouvelle topologie par leurs voisins immédiats uniquement. La propagation des informations se fait de proche en proche sous la forme d'une rumeur... on appelle cela un protocole de bavardage, gossiping en anglais.
- C et D au contact de la panne, réagissent immédiatement et tiennent compte de ce changement de topologie du réseau.

- Compte tenu des deux observations ci-dessus dans le bilan, on se doute que le routage global va être incohérent, ou va avoir de fortes chances de l'être puisque les nœuds A et B n'ont pas construit leur routage avec des tables de routage qui tenaient compte de la panne du lien C-D alors que C et D l'ont fait.
- Dans les tables de routage on s'aperçoit qu'il est suffisant d'échanger les informations de coût. Les informations de successeur sont reconstruites à chaque calcul assez naturellement.

### Question 3

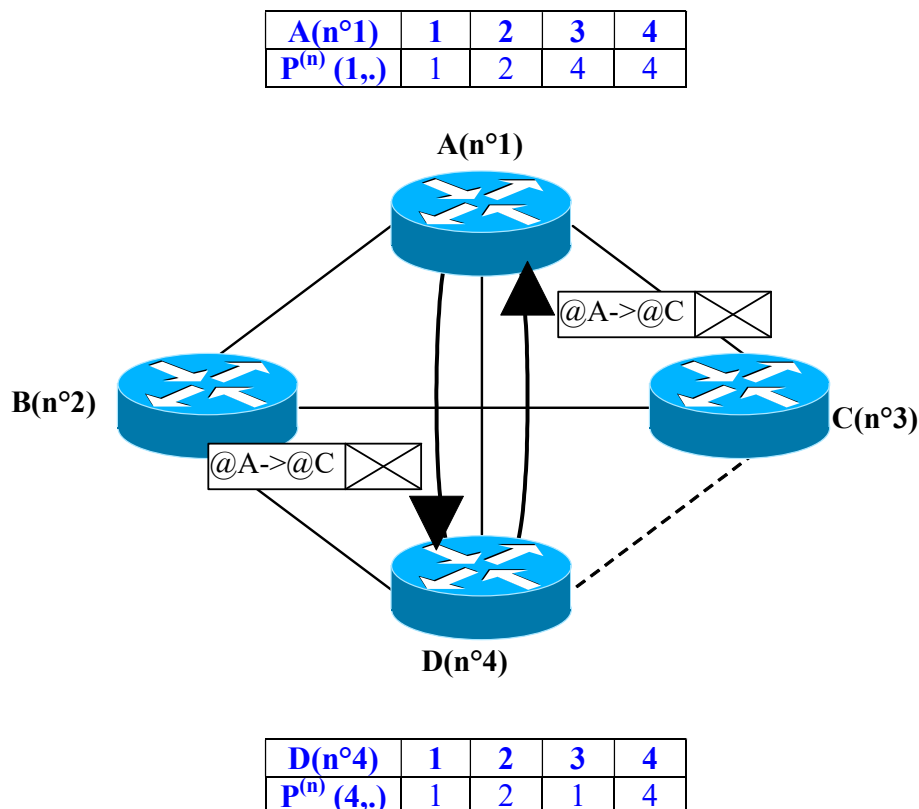
Pendant la première itération, un datagramme part d'un ordinateur hôte connecté au nœud de routage A (n°1) pour être transmis à un hôte connecté au nœud C (n°3).

Que se passe-t-il ? Décrivez le routage de ce datagramme.

#### Correction :

Le routeur A (n°1) souhaite envoyer un datagramme vers C (n°3). D'après sa table, le successeur est D. Arrivé sur D le datagramme doit rejoindre C mais la table des successeurs de D indique qu'il faut aller vers A... le datagramme va donc boucler indéfiniment entre D et A.

La figure ci-dessous à des fins de démonstration n'est pas tout à fait exacte par rapport au texte de la question posée. En effet, dans les datagrammes du dessin ce ne sont pas les adresses IP de A et C qui devraient apparaître mais celles des hôtes source et destination.





On voit ici l'intérêt du TTL dans le datagramme qui va se décrémenter au fur et à mesure de la traversée des routeurs et permettre d'éliminer le datagramme à la fin quand le TTL atteindra 0 si un routage d'ensemble ne se rétablit pas.

Phénomène intéressant on assiste à une capture de datagrammes, phénomène parfois baptisé aussi trou noir.

On va poursuivre la construction du routage avec le passage de l'étape  $n$  à l'étape  $n+1$ . On applique donc l'algorithme à nouveau. Au temps  $t_1$  les différents sites lancent une phase du calcul du routage optimal. Donc pour tout nœud  $i$  en considérant qu'on passe de l'étape  $n$  à l'étape  $n+1$  :

```
pour tout voisin  $k$  de  $i$ 
    Envoyer  $V^{(n)}(i, \bullet)$ ; %envoyer sa table de routage %
    Recevoir  $V^{(n)}(k, \bullet)$ ; %recevoir la table de routage de tous ses voisins
    Mesurer  $C^{(n+1)}(i, k)$ ;
finpour
```

Le calcul donne les résultats suivants pour l'étape  $n+1$ :

$V^{(n)}$	1	2	3	4	
1	0	4	2	1	$V(1,.)$
2	4	0	5	4	$V(2,.)$
3	4	8	0	5	$V(3,.)$
4	1	4	3	0	$V(4,.)$

C'est la matrice que tous les nœuds fabriquent parallèlement à partir de leur table et de celles reçues de leurs voisins

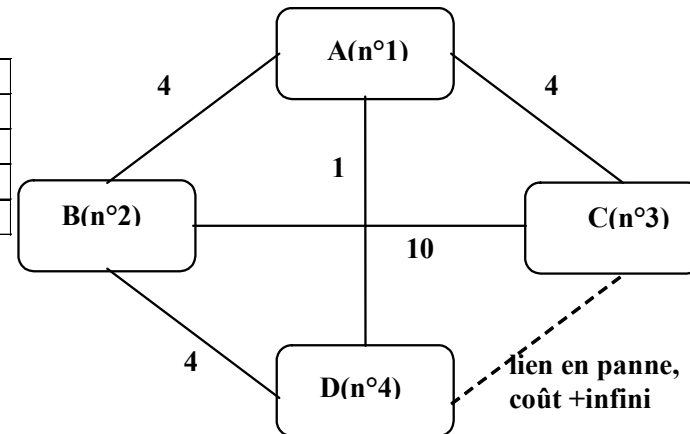
$C_1^{(n+1)}$	$S_1^{(n+1)}$	1	2	3	4
0	k=1	/	/	/	/
4	k=2	/	4	9	8
4	k=3	/	12	4	9
1	k=4	/	5	4	1

$C(n^0)$	1	2	3	4
$V^{(n+1)}(1,.)$	0	4	4	1
$P^{(n+1)}(1,.)$	1	2	3	4

2 choix : on prend celui qui est un lien direct en priorité

$C_2^{(n+1)}$	$S_2^{(n+1)}$	1	2	3	4
4	k=1	4	/	6	5
0	k=2	/	/	/	/
10	k=3	14	/	10	15
4	k=4	5	/	7	4

$C(n^0)$	1	2	3	4
$V^{(n+1)}(2,.)$	4	0	6	4
$P^{(n+1)}(2,.)$	1	2	1	4



$C_4^{(n+1)}$	$S_4^{(n+1)}$	1	2	3	4
1	k=1	1	5	3	/
4	k=2	8	4	9	/
+infini	k=3	+infini	+infini	+infini	/
0	k=4	/	/	/	/

$C(n^0)$	1	2	3	4
$V^{(n+1)}(4,.)$	1	4	3	0
$P^{(n+1)}(4,.)$	1	2	1	4

Pour les nœuds A(n°1) et B(n°2), l'effet de la panne du lien C-D est pris en compte à cette itération et influence l'élaboration des nouvelles tables de routage. Dans le schéma ci-dessus, pour le nœud D (n°4) on voit par contre que les calculs donnent des résultats identiques à ceux de l'itération n. Pour le nœud C (n°3), cela ne figure pas sur le schéma, mais le résultat est aussi identique à celui de l'itération n.

Si on faisait une itération n+2, les résultats ne changeraient plus : la rumeur de la panne a atteint tous les nœuds. Tous les nœuds "ont appris" l'état global du réseau.

Avec ces tables de routage, un datagramme pour C partant de A irait directement au bon routeur, c'est-à-dire C. La boucle n'apparaît plus, le routage est donc stabilisé.

La vitesse de convergence du calcul du routage optimum de RIP dépend de la taille du réseau, et de son degré de maillage. C'est l'un des reproches fait à RIP.

- **Pour le nœud A (n°1) voila les étapes :**

Il connaît :

A(n°1)	1	2	3	4
$P^{(n-1)}(1,.)$	1	2	4	4
$V^{(n-1)}(1,.)$	0	4	2	1

Il reçoit (dédit de la réponse à la question 1) :

B(n°2)	1	2	3	4
$P^{(n-1)}(2,.)$	1	2	4	4
$V^{(n-1)}(2,.)$	4	0	5	4

Il ne reçoit rien de C et donc A prend pour hypothèse que la table de routage de C devient :

C(n°3)	1	2	3	4
$P^{(n-1)}(3,.)$	?	?	?	?
$V^{(n-1)}(3,.)$	+infini	+infini	+infini	+infini

D(n°4)	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

On obtient alors pour résoudre le calcul du routage de A la matrice  $C^{(n)}(1,.)$  mesurée et la matrice  $V_1^{(n)}$

$C^{(n)}(1,.)$	1	2	3	4
1	0	4	4	1

$V_1^{(n-1)}$	1	2	3	4
1-A	0	4	2	1
2-B	4	0	5	4
3-C	+infini	+infini	+infini	+infini
4-D	1	4	1	0

i	$C^{(n)}(1,.)$	k pour $V_1^{(n-1)}$	1	2	3	4	$S_1^{(n)} j \rightarrow$	1	2	3	4
1	0	1	0	4	2	1	1	/	/	/	/
1	4	2	4	0	5	4	2	/	4	9	8
1	4	3	+infini	+infini	+infini	+infini	3	/	+infini	+infini	+infini
1	1	4	1	4	1	0	4	/	5	2	1

- Les résultats lors de la première réponse :
  - Matrice de coût des chemins  $V^{(n)}(1,.)$  :

$A(n^o1)$	1	2	3	4
$V^{(n)}(1,.)$	0	4	2	1

- Matrice des successeurs correspondante:

$A(n^o1)$	1	2	3	4
$P^{(n)}(1,.)$	1	2	4	4

- Les résultats avec la nouvelle hypothèse :
  - Matrice de coût des chemins  $V^{(n)}(1,.)$  :

$A(n^o1)$	1	2	3	4
$V^{(n)}(1,.)$	0	4	2	1

- Matrice des successeurs correspondante:

<b>A(n°1)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n)</sup> (1,.)</b>	1	2	4	4

On a donc le même résultat final.

- Pour le nœud B (n°2) voila les étapes :**

Il connaît :

<b>B(n°2)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (2,.)</b>	1	2	4	4
<b>V<sup>(n-1)</sup> (2,.)</b>	4	0	5	4

Il reçoit des autres nœuds :

<b>A(n°1)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (1,.)</b>	1	2	4	4
<b>V<sup>(n-1)</sup> (1,.)</b>	0	4	2	1

Même problème que pour A pour recevoir la table de routage de C qui doit passer par D, donc même hypothèse.

<b>C(n°3)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (3,.)</b>	?	?	?	?
<b>V<sup>(n-1)</sup> (3,.)</b>	+infini	+infini	+infini	+infini

<b>D(n°4)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (4,.)</b>	1	2	3	4
<b>V<sup>(n-1)</sup> (4,.)</b>	1	4	1	0

On obtient alors pour résoudre le calcul la matrice  $C^{(n)}(2,.)$  mesurée, et la matrice  $V_2^{(n-1)}$

<b>C<sup>(n)</sup> (2,.)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
2	4	0	10	4

$V_2^{(n-1)}$	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	+infini	+infini	+infini	+infini
4	1	4	1	0

On procède avec B(2) alors comme avec le nœud A(1).

i	$C^{(n)}$ (2,.)	k pour $V_2^{(n-1)}$	1	2	3	4	$S_1^{(n)} j \rightarrow$	1	2	3	4
2	4	1	0	4	2	1	1	4	/	6	5
2	0	2	4	0	5	4	2	/	/	/	/
2	10	3	+infini	+infini	+infini	+infini	3	+infini	/	+infini	+infini
2	4	4	1	4	1	0	4	5	/	5	4

On en déduit  $V^{(n)}(2, .)$  :

$B(n^o2)$	1	2	3	4
$V^{(n)}(2,.)$	4	0	5	4

La matrice des successeurs correspondante est:

$B(n^o2)$	1	2	3	4
$P^{(n)}(2,.)$	1	2	4	4

On constate que pour le nœud B (n°2) le changement d'hypothèse ne change pas le résultat.

• Pour le nœud C (n°3) voila les étapes :

Il connaît :

$C(n^o3)$	1	2	3	4
$P^{(n-1)}(3,.)$	4	4	3	4
$V^{(n-1)}(3,.)$	2	5	0	1

Il reçoit (dédit de la réponse à la question 1) :

<b>A(n°1)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (1,,)</b>	1	2	4	4
<b>V<sup>(n-1)</sup> (1,,)</b>	0	4	2	1

<b>B(n°2)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (2,,)</b>	1	2	4	4
<b>V<sup>(n-1)</sup> (2,,)</b>	4	0	5	4

Dans notre nouvelle hypothèse, il ne reçoit jamais la table de routage de D, et fait la même hypothèse que précédemment.

<b>D(n°4)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>P<sup>(n-1)</sup> (4,,)</b>	?	?	?	?
<b>V<sup>(n-1)</sup> (4,,)</b>	+infini	+infini	+infini	+infini

On obtient alors pour résoudre le calcul la matrice  $C^{(n)}(3,,)$  mesurée et la matrice  $V_3^{(n-1)}$

<b>C<sup>(n)</sup> (3,,)</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>3</b>	4	10	0	+infini

<b>V<sub>3</sub><sup>(n-1)</sup></b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>1</b>	0	4	2	1
<b>2</b>	4	0	5	4
<b>3</b>	2	5	0	1
<b>4</b>	+infini	+infini	+infini	+infini

i	C <sup>(n)</sup> (3,,)	k pour V <sub>3</sub> <sup>(n-1)</sup>	1	2	3	4	S <sub>1</sub> <sup>(n)</sup> j->	1	2	3	4
3	4	1	0	4	2	1	1	4	8	/	5
3	10	2	4	0	5	4	2	14	10	/	14
3	0	3	2	5	0	1	3	/	/	/	/
3	+infini	4	+infini	+infini	+infini	+infini	4	+infini	+infini	/	+infini

Le résultat ne change pas malgré le changement d'hypothèse.

On en déduit  $V^{(n)}(3, .)$  :

$C(n^{\circ}3)$	1	2	3	4
$V^{(n)}(3,.)$	4	8	0	5

La matrice des successeurs correspondante est :

$C(n^{\circ}3)$	1	2	3	4
$P^{(n)}(3,.)$	1	1	3	1

• **Pour le nœud D (n°4) voila les étapes :**

Il connaît sa table :

$D(n^{\circ}4)$	1	2	3	4
$P^{(n-1)}(4,.)$	1	2	3	4
$V^{(n-1)}(4,.)$	1	4	1	0

Il obtient de ses voisins :

$A(n^{\circ}1)$	1	2	3	4
$P^{(n-1)}(1,.)$	1	2	4	4
$V^{(n-1)}(1,.)$	0	4	2	1

$B(n^{\circ}2)$	1	2	3	4
$P^{(n-1)}(2,.)$	1	2	4	4
$V^{(n-1)}(2,.)$	4	0	5	4

Il ne reçoit pas la table de C et fait donc des hypothèses identiques à celles des autres nœuds pour cette situation :

$C(n^{\circ}3)$	1	2	3	4
$P^{(n-1)}(3,.)$	?	?	?	?
$V^{(n-1)}(3,.)$	+infini	+infini	+infini	+infini

On obtient alors pour résoudre le calcul la matrice  $C^{(n)}(4,.)$  mesurée et  $V_4^{(n-1)}$

$C^{(n)}(4,.)$	1	2	3	4
4	1	4	+infini	0



$V_4^{(n-1)}$	1	2	3	4
1	0	4	2	1
2	4	0	5	4
3	+infini	+infini	+infini	+infini
4	1	4	1	0

i	$C^{(n)}$ (4,.)	k pour $V_4^{(n-1)}$	1	2	3	4	$S_1^{(n)}$ j->	1	2	3	4
4	1	1	0	4	2	1	1	1	5	3	/
4	4	2	4	0	5	4	2	8	4	9	/
4	+infini	3	+infini	+infini	+infini	+infini	3	+infini	+infini	+infini	/
4	0	4	1	4	1	0	4	/	/	/	/

On en déduit  $V^{(n)}(4, .)$  :

$D(n^{\circ}4)$	1	2	3	4
$V^{(n)}(4,.)$	1	4	3	0

La matrice des successeurs correspondante est :

$D(n^{\circ}4)$	1	2	3	4
$P^{(n)}(4,.)$	1	2	1	4

Le changement d'hypothèse ne bouleverse pas le résultat final pour le nœud D ( $n^{\circ}4$ ).

Finalement, on en vient aux mêmes résultats qu'on reçoive ou non la table de routage de C ou pas.

## Éléments de mise en œuvre de RIP :

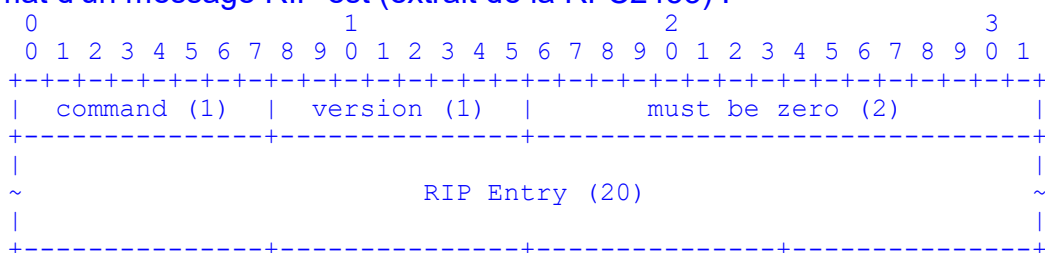
RIP V1 est classfull, il ne gère qu'une seule métrique, il compte le nombre de sauts (hop count). RIP V2 est classless et supporte plusieurs métriques. L'avantage de RIP sur OSPF son successeur est qu'il est simple, et facile à mettre en œuvre. Il peut donc être préféré sur un petit réseau pour faire du routage dynamique. La version la plus populaire de RIP correspond au programme routed inclu dans la distribution d'UNIX BSD.

Le routage RIP à une distance maximale de 15 noeuds entre tout routeur (diamètre du réseau), 16 noeuds indique que le réseau distant n'est pas accessible. La valeur d'un coût est donc comprise entre 1 et 15, 16 représente donc la valeur +infini avec laquelle nous avons fait l'exercice.

RIP V1 correspond au RFC1058. Initialement, le temps d'attente, entre deux calculs du routage, optimum était de 30 secondes. Un routeur envoie un datagramme de mise à jour, sur toutes ses interfaces, appelé "routing update". Pour cela, il utilise le protocole UDP (User Datagram Protocol) et le port 520. RIPv2 utilise le même port et le même protocole UDP.

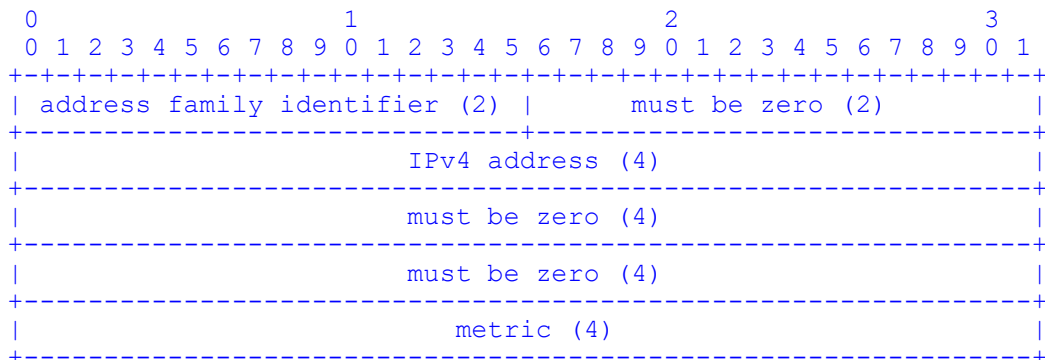
Cela fonctionne en client/serveur, le port serveur étant le 520. Les messages de mise à jour de tables partent du port 520 toutes les 30 secondes (valeur initiale). L'idée dans la mise en œuvre de RIP est d'essayer de rendre le déclenchement des mises à jour du routage asynchrone pour éviter des pointes de surcharges synchrones. Si aucun message de mise à jour n'est reçu pendant 180s le lien est considéré en panne et prend le coût 16 (+infini).

Le format d'un message RIP est (extrait de la RFC2453) :



La commande est soit une requête soit une réponse qui envoie tout ou partie de la table de routage de l'agent RIP sollicité.

Il peut y avoir entre 1 et 25 entrées RIP. Une entrée RIP V1 a le format suivant d'après la même RFC :



Les champs contiennent des informations au format big-endian. Ne pas oublier que le champ "metric" prend une valeur 1 à 15.



RIP V1 est vulnérable aux attaques car il n'a pas de mécanisme d'authentification. L'authentification pour RIP a été introduite en 1997 et utilise l'algorithme MD5 qui est un hashage cryptographique.

RIP V2 correspond au RFC 2453 auquel nous renvoyons le lecteur pour avoir plus de détails. Il corrige certaines faiblesses de RIP V1. Certaines fonctionnalités de RIP V2 permettent de maintenir une compatibilité avec RIP V1. RIPv2 étant classless donc compatible avec CIDR vu en cours, intègre le masque des réseaux qu'il gère dans ses tables de routage.

RIPv2 ne change pas le protocole RIP mais propose des extensions. Un descripteur d'entrée de la table de routage (Route Entry, RE) change :

Entries de la table de routage (Route Entry, RE) change :

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									
Address Family Identifier (2)																				Route Tag (2)																			
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									
										IP Address (4)																													
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									
										Subnet Mask (4)																													
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									
										Next Hop (4)																													
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									
										Metric (4)																													
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+										+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+									

On s'aperçoit que le masque apparaît dans le descripteur, et que cela ressemble plus aux tables de routage sur lesquelles nous avons travaillé lors des exercices des séances de TD précédentes.

Il apparaît un champ "Route Tag". Il permet de distinguer les routes internes à un domaine de routage (encore appelé Autonomous System) des routes externes. La valeur 0.0.0.0 dans le champ réseau indique que le routeur qui a émis le message est un routeur par défaut.

Il est possible, mais pas obligatoire, en RIPv2 d'envoyer les mises à jour de routage avec une adresse multicast-IP : 224.0.0.9.

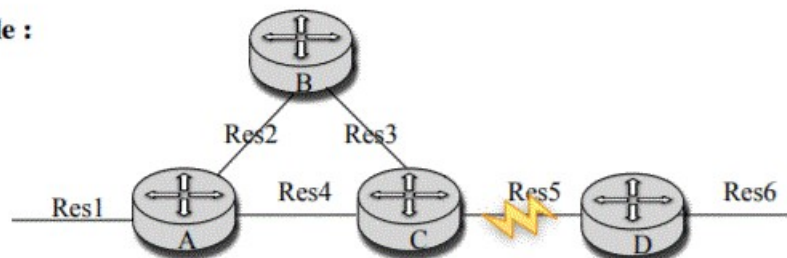
Je sou mets à votre sagacité la page wikipedia : [https://en.wikipedia.org/wiki/Distance-vector\\_routing\\_protocol#Count-to-infinity\\_problem](https://en.wikipedia.org/wiki/Distance-vector_routing_protocol#Count-to-infinity_problem) qui évoque le problème du comptage à l'infini avec RIP. C'est un autre exemple de routage RIP.

Un problème souvent évoqué lors de la mise en œuvre est le "comptage à l'infini". Dans la mesure où RIP est en voie d'obsolescence, on ne s'y attarde pas. Toutefois, le problème est bien décrit dans le cours de P. Sicard de l'Université Joseph Fourier à Grenoble, <http://lig-membres.imag.fr/sicard/crRES/cours%204%20Reseau.pdf> (consulté le 25/10/2020). Les planches du cours liées à ce problème sont extraites ci-après :



## Problème du comptage à l'infini (2)

- L'algorithme peut toujours être mis en défaut dans le cas de boucle dans le réseau
- **Exemple :**



**Si la ligne entre C et D est coupée, un scénario possible:**

C élimine de sa table de routage Res6 (plus de paquet RIP provenant de D)

Puis A élimine de sa table de routage Res6 car C ne lui envoie plus rien sur Res6

B n'a pas encore éliminer Res6 de sa table de routage (timer non synchronisé)

B envoie donc à A qu'il peut accéder à Res6 avec un coût de 3

Ensuite A va donc envoyer à C qu'il peut accéder à Res6 avec un coût de 4

Et ainsi de suite, on tourne en rond et à chaque paquet RIP le coût augmente de 1

## Solutions au comptage à l'infini

- Le problème survient dès que le réseau possède des boucles. Pour remédier complètement au problème il faudrait avoir une vision globale du réseau
- Le fait de limiter l'infini à un entier relativement petit limite les dégâts. Mais il doit être supérieure au nombre maximum de saut dans le réseau complet (RIP : 16)
- Pour éviter que les paquets tournent trop longtemps en rond, champ durée de vie de l'entête IP (TTL)

## Limitation du problème du comptage à l'infini

- **Pour augmenter la rapidité de la convergence en cas de panne ou de modification de topologie**
  - **Empoisonnement de route** (Route poisoning): A la sonnerie d'un timer le routeur ne supprime pas la ligne immédiatement mais lui associe un coût de 16 (inaccessible) qui sera propagé dans les prochains paquets RIP
    - Cette information est ainsi propagée plus rapidement mais n'évite pas le problème du "comptage à l'infini" dans tous les cas
  - **Mise à jour déclenchée**: En cas de modification de la table suite à la réception d'un paquet RIP, la modification est envoyée immédiatement.
    - Cela limite fortement la durée des comptages à l'infini

## Exercice 2 : Routage à état des liens, l'exemple OSPF (Open Shortest Path First) vu en UTC505.

Le but de ce problème est d'étudier la construction de la base de données d'état des liens utilisée dans le protocole internet OSPF. C'est un protocole de routage réparti dans lequel chaque routeur gère une base de données de l'ensemble des liaisons d'un réseau (topologie du réseau) et calcule à partir de cette base les plus courts chemins par l'algorithme de Dijkstra. Cette base de donnée est mise à jour par diffusion en inondation par chaque routeur de l'état de ses liaisons aux autres routeurs. Il entre dans la classe des protocoles baptisés "à état des liaisons" ou "linkstate".

### Question 1

Les protocoles de routage de type vecteurs de distance dont l'exemple type est le protocole Internet RIP ("Routing Information Protocol") se distinguent des protocoles de type Internet OSPF (à état de liaison).

1.1 Rappeler comment fonctionne RIP en quelques lignes ?

#### Correction :

Le principe de RIP est d'échanger **périodiquement** ses tables de routage avec ses voisins, puis d'effectuer un calcul de la table de routage locale à partir de ces nouvelles informations. Dans RIP, chaque routeur effectue l'algorithme :

```
cycle
  envoyer la table de routage d'un routeur avec les coûts connus de celui-ci
    à tous ses voisins immédiats
  recevoir la table de routage avec les coûts de tous les voisins immédiats
  calculer une nouvelle table de routage en prenant pour chaque destination
    le chemin qui correspond au minimum de tous les coûts possibles
    (en passant par les voisins immédiats).
fin cycle
```

Quelle est la différence fondamentale entre ces deux familles de protocoles de routage ?

#### Correction :

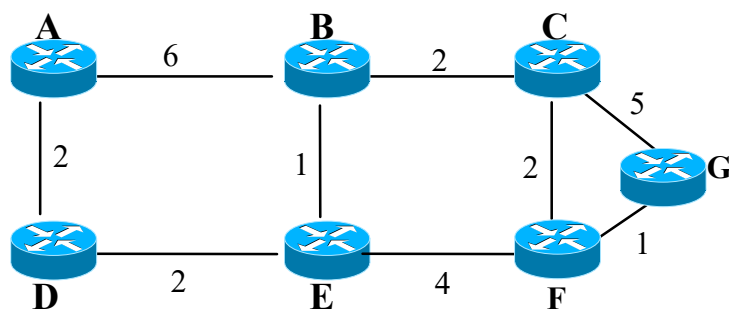
Les différences entre les algorithmes à état des liens et les algorithmes à vecteurs de distance sont reprises du cours et peuvent être résumées ainsi :



Etat des liens vs Vecteur de distances	
Dire à tous à propos de ses voisins	Dire à ses voisins à propos de tous
Inondation contrôlée de l'état des liens	Echange de vecteurs de distance avec ses voisins
Calcul par l'algo de Dijkstra	Calcul par l'algo de Bellman-Ford
Chaque routeur calcule sa propre table avec la connaissance de l'état de tous les liens	La table de chaque routeur est utilisée par les autres
Il peut y avoir des oscillations	Il peut y avoir des boucles
<b>Open Shortest Path First (OSPF)</b>	<b>Routing Information Protocol (RIP)</b>
Complexité en Messages	
$O(n^2 \cdot e)$ messages <ul style="list-style-type: none"> <li>• n: nb de noeuds</li> <li>• e: nb d'arcs</li> </ul>	$O(d \cdot n \cdot k)$ messages <ul style="list-style-type: none"> <li>• d: nb de degrés d'un noeuds</li> <li>• k: nb de "rounds"</li> </ul>
Complexité en temps	
$O(n \cdot \log n)$	$O(n)$
Convergence en temps	
$O(1)$	$O(k)$
Robustesse: que se passe-t-il si un routeur dysfonctionne?	
<ul style="list-style-type: none"> <li>– Chaque noeud peut annoncer un lien incorrect avec son coût</li> <li>– Chaque noeud calcule sa propre table</li> </ul>	<ul style="list-style-type: none"> <li>– Chaque noeud peut annoncer un chemin incorrect via son coût</li> <li>– La table de chaque noeud est utilisée par les autres; les erreurs se propagent dans le réseau</li> </ul>

## Question 2

Par exemple soit un réseau de 7 routeurs A, B, C, D, E, F, G dont la topologie est la suivante (les coûts de transit pour chaque liaison sont supposés égaux dans chaque direction et sont mentionnés sur l'arc représentant la liaison).



La base de données (topologie ou carte du réseau) qui doit être connue de chaque routeur donne principalement les coûts en point à point pour chaque liaison. D'autres informations sont également stockées dans cette table. Certaines de ces informations seront introduites dans la suite.

De	Vers	Coût
A	B	6
A	D	2
B	A	6
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	2
E	B	1
E	D	2
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

Cette base de données est construite par échange d'informations entre les routeurs. Pour cela le protocole suivant est effectué. Son déclenchement peut répondre à différentes stratégies de mise à jour :

- périodiquement
- lorsqu'un routeur nouveau s'initialise
- lorsqu'un routeur s'aperçoit qu'il a un nouveau voisin
- lorsque le coût d'une liaison avec un voisin a changé

• Étape 1 :

Chaque routeur construit un paquet appelé "paquet d'état des liaisons" ou LSP ("Link State Packet") qui contient des coûts de liaison que le routeur souhaite faire connaître. Un LSP comporte principalement une liste de noms de routeurs (voisins d'un routeur) et les coûts pour les atteindre. Ces champs sont appelés LSA (Link State Advertisements). Les LSP émis par un même routeur sont numérotés au moyen d'un numéro de séquence. Pour simplifier on ne se préoccupe pas du retour à zéro des compteurs utilisés trop longtemps.

• Étape 2 :

Le paquet LSP est transmis à tous les routeurs voisins et chaque routeur enregistre les informations du LSP généré le plus récemment. Plus précisément chaque voisin effectue le traitement suivant:

1. Recevoir le paquet LSP.
2. Consulter la base existante.
3. Si l'entrée (la liaison et son coût) n'est pas présente, ajouter cette entrée et



diffuser l'information à tous les voisins sauf l'émetteur du LSP.

4. Si l'entrée est présente et si le numéro de séquence du LSP est plus grand que celui correspondant à l'entrée modifier l'entrée et diffuser l'information à tous les voisins sauf le ré-émetteur du LSP.
5. Si l'entrée est présente et si le numéro de séquence du LSP est plus petit ou égal à celui correspondant à l'entrée: ne rien faire.

A quoi pourrait servir la retransmission du LSP à son émetteur sachant que OSPF envoie ses LSP au dessus du protocole IP ?

### Correction :

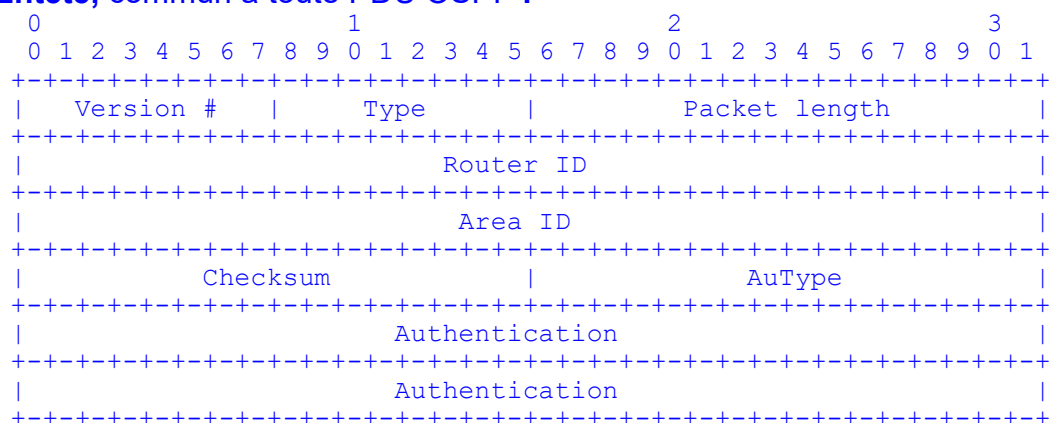
Rappelons que OSPF utilise IP pour transmettre ses informations de service, et que donc les transmissions ne sont pas fiables puisqu'IP est un service à datagramme. Le numéro de protocole correspondant dans le datagramme IP est 89.

L'émetteur connaît l'information contenue dans le paquet LSP puisqu'il vient juste de l'envoyer. D'un point de vue connaissance de l'état du réseau, la retransmission d'un paquet LSP est donc inutile. Par contre la retransmission vers l'émetteur pourrait permettre de savoir que le paquet a été correctement reçu par son destinataire. On retrouverait alors un protocole de type écho. Cela pourrait donc avoir un certain sens puisque OSPF envoie ses annonces LSP directement au-dessus de IP.

Mais la forme d'acquiescement positif énoncée ci-dessus est coûteuse car le paquet peut être volumineux et la comparaison avec le paquet émis est longue. Il est préférable si l'on veut un acquiescement positif de générer un message court ayant cette signification.

Format d'un LSU, Link State Update (anciennement LSP, sigle que nous utilisons dans l'exercice), d'après la RFC 2328 (<http://www.rfc-editor.org/rfc/rfc2328.txt>) :

- **Entête**, commun à toute PDU OSPF :



Version : numéro de version d'OSPF

Type nature de la PDU OSPF :

- |   |                           |
|---|---------------------------|
| 1 | Hello                     |
| 2 | Database Description      |
| 3 | Link State Request        |
| 4 | Link State Update         |
| 5 | Link State Acknowledgment |

Packet length : longueur de toute la PDU OSPF

Router ID : ID du routeur source

Area ID : ID de l'AREA sur 32 bits

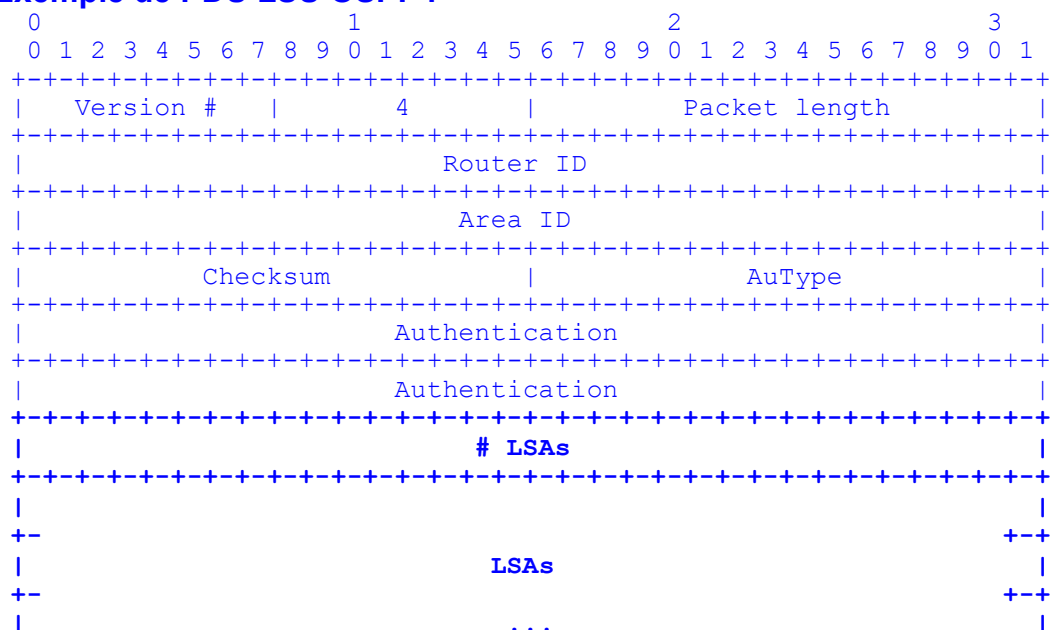
Checksum : CRC sur le PDU OSPF complet sauf les 64 bits d'authentification

AuType : identifie la procédure d'authentification utilisée pour la PDU OSPF

Authentication : champ d'authentification sur 64 bits



- Exemple de PDU LSU OSPF :



Pourquoi un paquet LSP n'est-il pas renvoyé à son émetteur du point de vue du mécanisme d'inondation ?

**Correction :**

La retransmission est dangereuse car en l'absence de mécanisme de contrôle de l'inondation l'émetteur pourrait rediffuser la même information (principe de l'inondation) et on aurait une explosion combinatoire incontrôlable de messages dans le réseau qui le saturerait entraînant une congestion des routeurs.

### Question 3

A quoi sert le Numéro de Séquence émetteur du point de vue du réseau internet qui est à datagramme ?

**Correction :**

Un réseau à datagramme comme IP peut déséquencer et perdre les paquets. Il n'y a pas de contrôle de séquence et de contrôle d'erreur effectué. Le numéro de séquence peut permettre de construire pour les besoins du protocole de routage seulement ces deux fonctions. En particulier si deux paquets LSP sont déséquencés (l'un dépasse l'autre) l'information la plus à jour (la première arrivée) sera seule enregistrée.

A quoi sert le Numéro de Séquence émetteur du point de vue du protocole de diffusion des informations d'état de liaison ?

**Correction :**

Le protocole à diffusion en inondation en l'absence d'un mécanisme de contrôle de l'inondation génère un nombre exponentiel de messages. Ici le numéro de séquence permet à chaque routeur de ne transmettre en inondation qu'une seule fois une information, en fait l'information la plus à jour. Ainsi un même paquet ne circule dans le pire des cas que deux fois sur une même liaison.

Le numéro de séquence joue un rôle de numéro de version en quelque sorte.



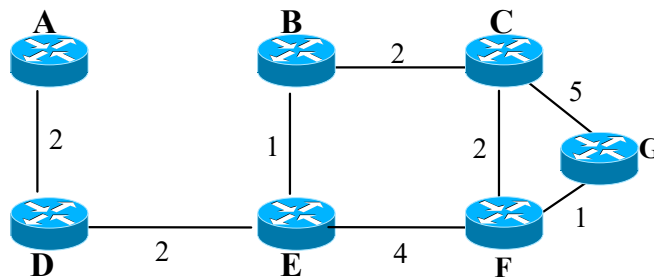
En plus du Numéro de Séquence, chaque information concernant une liaison possède une date de péremption (variable baptisée dans OSPF "age") qui apparaît aussi bien dans les paquets LSP échangés que dans les bases de données. Toute information dépassant sa date est systématiquement détruite. Citer plusieurs types de problèmes qu'une telle datation solutionne.

**Correction :**

Typiquement les dates de péremption permettent de traiter les changements inopinés de configuration (mobilité non détectée d'un appareil). Également de nombreux types d'erreurs pourront être traités ainsi. Supposons qu'une valeur de coût soit fausse (erreur non corrigée de transmission, ou erreurs répétées non détectées). Cette valeur ne sera pas utilisée plus longtemps que la date de péremption. La date de péremption dans un paquet LSP inondé permet d'être certain qu'il n'existe pas de paquets de mise à jour qui "tournent" indéfiniment.

**Question 4**

On suppose que la liaison de A vers B est coupée. On a alors le réseau :



A ayant détecté la coupure, A prépare un LSP de la forme : "de A à B, coût = infini, numéro de séquence,"

On suppose que ce LSP est le premier généré (au début il est le seul).

Indiquer ce qui se passe alors dans le réseau. Quelle est la base de données obtenue par les différents noeuds à la fin du processus d'inondation ?

**Correction :**

- D va envoyer à E ce LSP et E va corriger toutes les entrées de sa table correspondant aux valeurs contenues dans ce LSP. Il n'y a qu'une seule entrée ici. D ne renvoie pas ce LSP à A (qui est l'émetteur de ce LSP) et D obtient la base :

De	vers	coût
A	B	infini
A	D	2
B	A	6
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	2
E	B	1
E	D	2
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

qui est aussi la base que possède A.

- D va envoyer à E le même LSP constitué des lignes qui ont changées (ici de la ligne qui a changée) dans sa base à E c'est à dire le LSP "de A à B coût = infini"
- E va procéder de même, obtenir la même base et envoyer ce même LSP aux voisins B et F mais pas à D qui en est l'émetteur.
- B et F vont faire de même, obtenir la même base et B va envoyer ce même LSP à C (et pas à E qui est l'émetteur), F va l'envoyer à C et G uniquement.
- C va bien recevoir 2 fois le même LSP de 2 voisins distincts, mais ces 2 LSP ont un contenu identique et il va obtenir la même base. Il enverra cet LSP uniquement à G qui lui aussi en recevra ainsi 2 (un provenant de C et l'autre de F), qui sont identiques. Donc G obtiendra la même base et le processus d'inondation s'arrête.

En résumé le processus d'inondation s'arrête au bout d'un temps fini en ayant fait passer sur chaque liaison un et un seul LSP et tous les noeuds ont élaboré la même base qui est celle ci dessus.

### Remarque :

La matrice ci dessus n'est pas symétrique (valeurs distinctes pour aller de A vers B et pour aller de B vers A) mais c'est parce que nous n'avons pas étudié le cas où B envoie le LSP "de B à A coût = infini". L'inondation de cet LSP rend la base symétrique

On suppose qu'ensuite B prépare et transmet son LSP concernant A-B.

Quelle est la base de données obtenue ?

### Correction :

Suivant le même principe que ci-dessus, B va déclencher un processus d'inondation pour que les routeurs du réseau puisse mettre à jour leur base de données des liens. Il va envoyer un le LSP "de B à A coût = infini".

Il faut bien avoir en tête qu'OSPF travaille avec des arcs et non des liens. Chaque sens d'un lien compte pour une entrée dans la base de données de l'état des liens.



La base de données de topologie calculée et commune à tous les routeurs après que B ait signalé la rupture est la suivante:

De	vers	coût
A	B	infini
A	D	2
B	A	infini
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	2
E	B	1
E	D	2
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

### Question 5

On suppose que la liaison de D à E tombe en panne (on suppose que les 2 noeuds D et E se sont aperçus de cette rupture et envoient les LSP correspondants).

Indiquer la (ou les) bases de données obtenues à l'issue de l'algorithme distribué de transmission de l'état des liaisons par chacun des noeuds.

#### **Correction :**

Après la coupure E-D le réseau est déconnecté, deux composantes connexes apparaissent.

- D va générer le LSP "de D à E cout = infini" qu'il enverra à A. Les deux sites A et D obtiennent la même table de routage :

De	vers	coût
A	B	infini
A	D	2
B	A	infini
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	infini
E	B	1
E	D	2
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

et le processus s'arrête là.

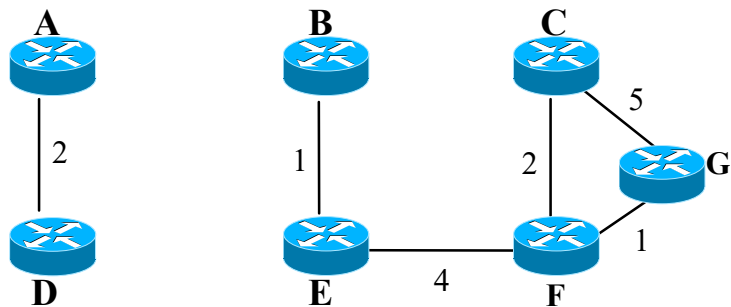
- Par contre, après inondation dans la partie "B, C, E, F, G" du réseau ces noeuds obtiennent tous une même table de routage qui est :

De	vers	coût
A	B	infini
A	D	2
B	A	infini
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	2
E	B	1
E	D	infini
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

On remarque que ces 2 tables ne sont pas identiques ni symétriques. Mais elles sont uniques à l'intérieur de chaque sous réseau connexe et c'est l'essentiel.

Question 6

A partir des bases de données trouvées à la question précédente et du réseau suivant on suppose que la liaison de B à C tombe en panne et amène au réseau :



Indiquer les bases de données obtenues par chaque routeur.

Correction :

- Tout le "sous réseau" connexe B, C, E, F, G sera informé par inondation de cette rupture de liaison et ces noeuds finiront par avoir la base :

De	vers	coût
A	B	infini
A	D	2
B	A	infini
B	C	infini
B	E	1
C	B	infini
C	F	2
C	G	5
D	A	2
D	E	2
E	B	1
E	D	infini
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

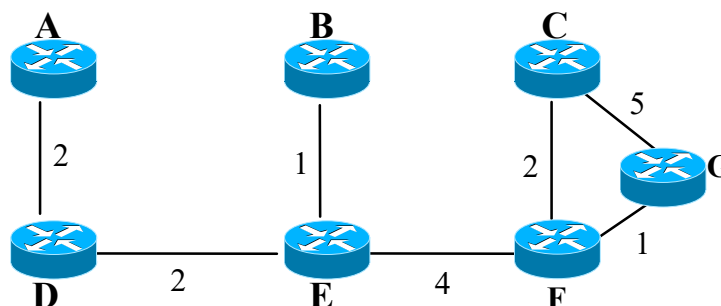
Par contre les noeuds A et D ne pourront être informés et conserveront la table qu'ils avaient auparavant :

De	vers	coût
A	B	infini
A	D	2
B	A	infini
B	C	2
B	E	1
C	B	2
C	F	2
C	G	5
D	A	2
D	E	infini
E	B	1
E	D	2
E	F	4
F	C	2
F	E	4
F	G	1
G	C	5
G	F	1

### Question 7

Par la suite la liaison de D à E est rétablie avec un coût de 2 rendant de nouveau le réseau connexe. On obtient alors le réseau ci-dessous.

On exécute l'algorithme de diffusion en supposant que les nœuds, D et E, informent le réseau du rétablissement de cette connexion.



Quelle est la valeur du coût de la liaison de B à C pour les sites A, D et pour les sites B, C, E, F ?

### Correction :

Lors du rétablissement de la connexion il faut évidemment que D et E envoient à leur voisin les LSP "de D à E coût = 2". Mais cela ne suffit pas pour que les bases de données deviennent cohérentes. Les bases de A et D indiquent un coût de 2 pour aller de B à C alors que celles des noeuds B, E, C, F et G indiquent un coût infini ce qui est la réalité. Il faut donc trouver un moyen pour amener les bases de données à être cohérentes et exactes.



Quel est le risque encouru dans cette situation ?

**Correction :**

Il faut absolument pour que le routage reste correct qu'une procédure de "mise en cohérence" permette de déterminer les valeurs les plus récentes des coûts pour faire converger les différentes copies des bases de données. Nous sommes dans un schéma de base de données répliquées. Sinon des paquets vont être mal routés et éventuellement ils vont cycler dans le réseau.

Comment détecter efficacement la situation ? Sur quel type de site ? On suggère une utilisation du numéro de séquence.

**Correction :**

Le numéro de séquence permet de savoir quelle est la valeur correcte (ici l'infini) car ce numéro qui correspond à la valeur la plus récente est nécessairement le plus grand. Le numéro de séquence offre un moyen de datation unique et cohérent des modifications. Une valeur de coût d'une liaison ne peut avoir été modifiée que par son site adjacent d'origine et chaque modification successive porte un numéro unique croissant.

Deux sites voisins qui possèdent deux tables non cohérentes comme D et E sont les mieux placés pour s'apercevoir de la divergence des bases de données et déclencher un protocole de mise en cohérence.

Comment faire pour mettre en oeuvre une solution de "réconciliation" basée sur le numéro de séquence ?

**Correction :**

Il faut transmettre les bases de données entre les deux anciens sous réseaux pour assurer la "réconciliation" entre les valeurs ayant divergé. Ici le seul problème est un problème d'efficacité. C'est une partie du protocole OSPF appelé "établissement de voisinage" qui se charge de ce travail. Un site active ce protocole pour détecter les divergences. Il demande et reçoit la table. Par comparaison il détecte les divergences. Il communique alors les entrées à mettre à jour par le mécanisme d'inondation habituel.

## Question 8

La technique de diffusion par inondation de l'état des liaisons est inconcevable telle que pour le réseau Internet. Pourquoi ?

**Correction**

L'inondation génère un très grand nombre de messages, et si le nombre de nœuds participants au routage OSPF est très grand on risque de charger considérablement le réseau par du trafic de service. Ce n'est pas envisageable.

De plus, si un réseau est très grand comme internet la base de données à traiter est énorme (les routeurs ne peuvent matériellement la stocker sauf avoir des configurations gigantesques avec un risque qui est qu'en raison de l'extension du réseau aucune machine ne puisse plus contenir la table) et les informations à acheminer sont trop conséquentes. (la diffusion encombre trop le réseau).

Que fait-on pour appliquer quand même OSPF à l'Internet ?

**Correction :**

Pour éviter ces problèmes, on hiérarchise le réseau ce qui a pour conséquence de hiérarchiser le routage. Le réseau est administrativement découpé selon différents niveaux. Les domaines ou "autonomous systems" constituent des entités administrées sous une autorité unique (par exemple une entreprise, un campus). On distingue les routages intra-domaines et inter-domaines. Le routage inter-domaines administratifs (entités économiques, gouvernementales, académiques...) est réalisé par BGP (Border Gateway Protocol)

Le protocole OSPF est destiné au routage intra-domaine. Cependant un domaine peut être gros voire très gros. Certaines sociétés multinationales gèrent des dizaines de milliers de sites avec un très grand nombre de routeurs. Pour améliorer la hiérarchisation OSPF permet le découpage des domaines en régions. Une région (area) est un sous-réseau d'une taille permettant l'application du principe de routage à état de liaison comme nous l'avons étudié. Chaque région se comporte comme un réseau presque indépendant et le protocole d'inondation s'arrête aux noeuds de cette zone. Pour communiquer entre les régions, OSPF utilise une région particulière: épine dorsale (backbone) sur laquelle les régions se connectent et qui permet la communication inter régions, c'est la région numéro 0.

**Question 9 :**

Lorsqu'un routeur X possède une base de données de topologie, il calcule le coût minimal d'un routeur à un autre ainsi que le routage à effectuer pour relier ces routeurs en utilisant l'algorithme de Dijkstra (1959).

Pour cet algorithme on utilise :

- PATH ensemble de couples (nom du noeud, coût du chemin du routeur X à ce noeud). Cet ensemble sera construit de proche en proche dans l'algorithme et indique à chaque instant les noeuds pour lesquels on a trouvé un meilleur chemin en partant de X.
- TENT est un ensemble de couples similaires à PATH. TENT signifie tentative et est un ensemble auxiliaire de calcul pour l'algorithme. Lors de l'algorithme les couples sont d'abord déposés dans TENT. Lorsqu'un chemin trouvé est effectivement le meilleur il passe de l'ensemble TENT à l'ensemble PATH.
- Un arbre de routage ayant pour racine le noeud X est construit de proche en proche lors de l'algorithme. On met en fait dans cet arbre les doublets décrits ci dessus.

L'algorithme de Dijkstra pour un routeur X est :

- Phase 1 - mettre dans PATH la valeur (X, 0) et comme racine de l'arbre cette valeur. C'est une phase d'initialisation.
- Puis on boucle :
  - Phase 2 - Pour le noeud Y qui vient d'être mis dans PATH examiner dans la base de données d'état des liens, les LSP d'origine Y.  
Pour chaque voisin Z de Y non présent dans PATH, ajouter le coût du trajet de X à Y au coût de Y à Z. On obtient les doublets (Z, nouveau coût).  
Placer ces doublets dans l'arbre de routage à la suite du doublet (Y, ...).  
Ne garder dans l'arbre qu'un seul exemplaire de chaque noeud présent : celui de coût associé minimum.



Si Z ne figure ni dans PATH, ni dans TENT avec un meilleur coût, ajouter le doublet (Z, nouveau coût) à TENT.

- Phase 3 - Si TENT est vide l'algorithme est terminé. Sinon on cherche dans TENT le doublet (ID, coût) où la valeur de coût est minimale. On fait passer ce doublet de TENT à PATH et on se reporte à la phase 2.

**Remarque :** La phase 3 consiste à rechercher le noeud de coût minimal et de le mettre dans PATH avant les autres. C'est ce qui donne le nom à cet algorithme: Shortest Path First (le chemin le plus court d'abord).

### 9.1 Pourquoi l'algorithme converge t il ?

#### **Correction :**

Les éléments ajoutés dans TENT ont un coût toujours plus petit. Pour chaque noeud du réseau, la suite du coût de X à ce noeud mis dans TENT est décroissante minorée par 0 donc convergente donc ce noeud apparaît un nombre fini de fois dans TENT et l'ensemble des noeuds est fini. Comme TENT perd à chaque fois un élément il finit par devenir vide et l'algorithme s'arrête.

**9.2** Décrivez le fonctionnement de l'algorithme dans ses différentes étapes pour le noeud A du réseau initial décrit à la question 2. On précisera les valeurs des ensembles PATH, TENT et de l'arbre de routage ainsi que l'arbre de routage final.

#### **Correction :**

On a les étapes suivantes :

- étape 1 :

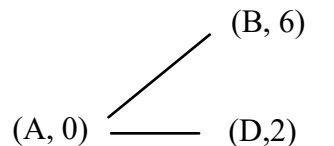
- **phase 1**

arbre : (A, 0)

PATH = {(A, 0)}

- **phase 2**

Les voisins de A sont B et D et on construit les doublets (B, 6) et (D, 2). On place ces doublets dans l'arbre :



L'ensemble TENT vaut {(B, 6), (D, 2)}.

- **phase 3**

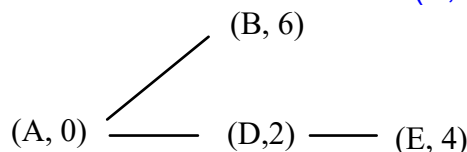
Un doublet où le coût est de valeur minimale est (D, 2). On choisit ce doublet à faire passer dans PATH. PATH vaut alors {(A, 0), (D, 2)} et TENT = {(B, 6)}

- étape 2

C'est le sommet D qui vient d'être mis dans PATH.

- **phase 2**

Les voisins de D sont A (qui est dans PATH) et E. On ne construit que le doublets (E, 4). On place ce doublets dans l'arbre à la suite de (D,2) :



L'ensemble TENT vaut {(B, 6), (E, 4)}.

- **phase 3**

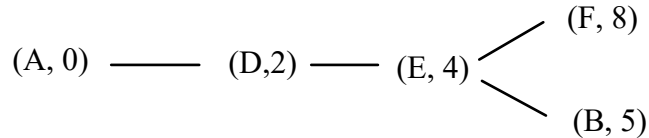
Le doublet où le coût est de valeur minimale est (E, 4). Il passe alors dans PATH et on obtient :  $PATH = \{(A, 0), (D, 2), (E, 4)\}$  et  $TENT = \{(B, 6)\}$ .

- étape 3

C'est le sommet E qui vient d'être mis dans PATH.

- **phase 2**

Les voisins de E sont D (qui est déjà dans PATH), B et F et on construit les doublets (B, 5) et (F, 8). On place ces doublets dans l'arbre après (E,4) et on supprime le doublet (B, 6) de coût supérieur au doublet (B, 5) :



L'ensemble TENT vaut  $\{(B, 5), (F, 8)\}$

- **phase 3**

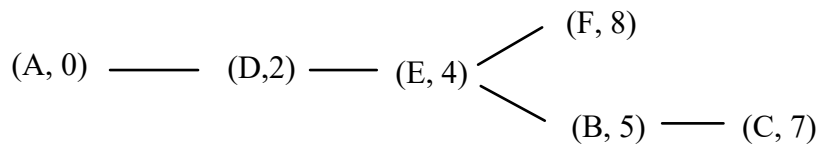
Le doublet où le coût est de valeur minimale est (B, 5). Il passe alors dans PATH et on obtient :  $PATH = \{(A, 0), (D, 2), (E, 4), (B, 5)\}$  et  $TENT = \{(F, 8)\}$

- étape 4

C'est le sommet B qui vient d'être mis dans PATH.

- **phase 2**

Les voisins de B sont A, E (qui sont dans PATH) et C et on construit le doublet (C, 7). On place ce doublet dans l'arbre :



L'ensemble TENT vaut  $\{(C, 7), (F, 8)\}$ .

- **phase 3**

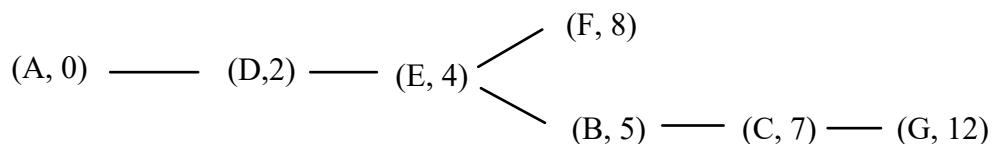
Le doublet où le coût est de valeur minimale est (C, 7). Il passe alors dans PATH et on obtient :  $PATH = \{(A, 0), (D, 2), (E, 4), (B, 5), (C, 7)\}$  et  $TENT = \{(F, 8)\}$ .

- étape 5

C'est le sommet C qui vient d'être mis dans PATH.

- **phase 2**

Les voisins de C sont B (qui est dans PATH), F et G et on construit le doublet (F, 9) et (G, 12). (F, 9) est de coût supérieur à (F, 8) donc seul le doublet (G, 12) est mis dans l'arbre :



L'ensemble TENT vaut  $\{(G, 12), (F, 8)\}$

- **phase 3**

Le doublet où le coût est de valeur minimale est (F, 8). Il passe alors dans PATH et on obtient :  $PATH = \{(A, 0), (D, 2), (E, 4), (B, 5), (C, 7), (F, 8)\}$  et  $TENT = \{(G, 12)\}$ .

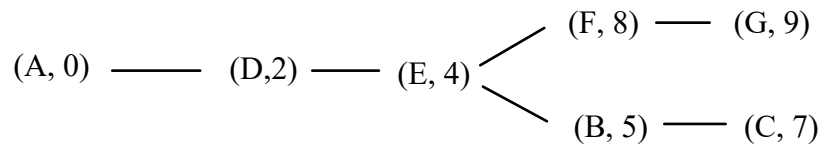
- étape 6

C'est le sommet F qui vient d'être mis dans PATH.

- **phase 2**



Les voisins de F sont E, C (qui sont dans PATH) et G. On construit le doublet (G, 9). (G, 9) est de coût inférieur à (G, 12) donc le doublet (G, 12) est supprimé de l'arbre et on obtient :



L'ensemble TENT vaut {(G, 9)}

○ **phase 3**

Le doublet où le coût est de valeur minimale est (G, 9). Il passe alors dans PATH et on obtient : PATH = {(A, 0), (D, 2), (E, 4), (B, 5), (C, 7), (F, 8), (G, 9)} et TENT est vide.

• étape 7

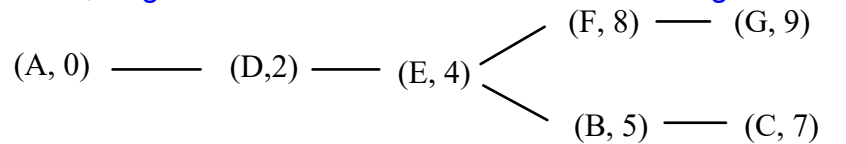
C'est le sommet G qui vient d'être mis dans PATH.

○ **phase 2**

Les voisins de G sont F et C (qui sont dans PATH) et il n'y a donc pas de doublet à construire ni à ajouter.

○ **phase 3**

TENT étant vide, l'algorithme est terminé est l'arbre de routage du noeud A est :



## Exercice 3 : Eléments de mise en oeuvre du protocole OSPF

Voilà une trace d'un échange OSPF :

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.23.2	224.0.0.5	OSPF	78	Hello Packet
4	9.995607	192.168.23.2	224.0.0.5	OSPF	78	Hello Packet
9	19.994741	192.168.23.2	224.0.0.5	OSPF	78	Hello Packet
13	29.993634	192.168.23.2	224.0.0.5	OSPF	78	Hello Packet
16	39.993236	192.168.23.2	224.0.0.5	OSPF	78	Hello Packet
17	40.081276	192.168.23.1	224.0.0.5	OSPF	78	Hello Packet
20	49.991592	192.168.23.2	224.0.0.5	OSPF	82	Hello Packet
21	50.078804	192.168.23.1	224.0.0.5	OSPF	82	Hello Packet
22	50.082034	192.168.23.2	192.168.23.1	OSPF	66	DB Description
30	55.079299	192.168.23.2	192.168.23.1	OSPF	66	DB Description
33	59.990374	192.168.23.2	224.0.0.5	OSPF	82	Hello Packet
34	59.998455	192.168.23.2	192.168.23.1	OSPF	206	DB Description
35	60.010935	192.168.23.2	192.168.23.1	OSPF	66	DB Description
36	60.020925	192.168.23.2	192.168.23.1	OSPF	66	DB Description
37	60.077775	192.168.23.1	224.0.0.5	OSPF	82	Hello Packet
38	60.522191	192.168.23.2	224.0.0.5	OSPF	202	LS Update
39	60.532700	192.168.23.1	224.0.0.5	OSPF	146	LS Update
42	63.030024	192.168.23.1	224.0.0.5	OSPF	98	LS Acknowledge
43	63.038301	192.168.23.2	224.0.0.5	OSPF	78	LS Acknowledge
47	69.989377	192.168.23.2	224.0.0.5	OSPF	82	Hello Packet

Election d'un routeur Maître (Designated Router- DR) 192.168.23.2, et d'un routeur Maître de secours (Backup Designated Router - BDR) 192.168.23.1 sur le réseau auxquels les deux interfaces appartiennent.

Synchronisation des Bases de Données d'Etat des liens à l'initiative du DR 192.168.23.2 avec 192.168.23.1

### Question 1

A quelle périodicité une interface envoie un paquet de type Hello d'après la trace ci-dessus. Arrondir la valeur à la dizaine de secondes la plus proche. Est-ce que cette période est respectée pendant toute la trace ? Expliquer brièvement.

#### Correction :

On regarde les Hello Packet, qui sont indiqués dans la colonne la plus à gauche. Entre la trame 1 la trame 4 il s'écoule quasiment 10 secondes si on fait la soustraction entre les deux dates d'affichage des trames.. Entre la trame 4 et la 9, pareil, entre 9 et 13 aussi...c'est



quasiment régulier. Attention, le trame 17 correspond à un autre routeur, et son Hello Packet suivant en ligne 21 est 10 secondes après. Entre ligne 20 et ligne 33, pour 198.63.23.2 encore 10 secondes d'écart. On peut en déduire que la périodicité est de 10 secondes.

## Question 2

Dans la première trame qui contient un message OSPF, l'adresse Ethernet destination est 01:00:5e:00:00:05. Cette adresse désigne-t-elle :

- Une adresse MAC d'une interface Ethernet destination encore appelée adresse unicast,
- Une adresse MAC correspondant à un broadcast Ethernet,
- Une adresse MAC correspondant à un multicast Ethernet ?

Pourquoi ?

### Correction :

A cause du 1 dans le premier octet, c'est une adresse multicast. Cf figure ci-dessous avec les informations entourées en bleu.

Une adresse broadcast serait "tous les bits à 1", soit FF:FF:FF:FF:FF:FF.

Une adresse unicast serait "00" dans le premier octet de l'adresse MAC, pourvu que ça soit une adresse constructeur.

Autre information qu'on peut déduire dans la trace, comme dans la figure ci-dessous, c'est l'adresse IP en 224.x.x.x, le préfixe 224. est associé à des adresses multicast. En particulier, 224.0.0.5 est réservée pour OSPF, c'est une adresse qui cible comme destination tous les routeurs OSPF du sous-réseau. Ces datagrammes ont d'ailleurs un TTL de 1 pour ne pas sortir du sous-réseau.

A titre indicatif, RIPv2 a l'adresse IP multicast réservée 224.0.0.9.

## Question 3

On rappelle que l'organisme IEEE, qui a normalisé Ethernet sous la référence IEEE802.3, représente les bits d'un octet dans l'ordre inverse de celui adopté par l'organisme de standardisation Internet qu'est l'IETF. En conséquence le bit qui indique dans une adresse MAC que c'est une adresse de diffusion n'occupe pas la même place en fonction qu'on lit une adresse Ethernet en mode IEEE802.3 ou en mode IETF.

Wireshark affiche les bits d'un octet d'une adresse MAC comme l'IETF ou comme l'IEEE ?

### Correction :

En première intuition, Wireshark colle au modèle Internet, on peut faire l'hypothèse que Wireshark affiche les adresses comme dans les documents de l'IETF, et donc qu'il numérote les bits d'un octet de la même façon de la droite vers la gauche, en big endian.

Le plus simple, c'est de voir, dans le détail une trace Wireshark jusqu'au niveau bit dans une adresse MAC pour en avoir le cœur net.

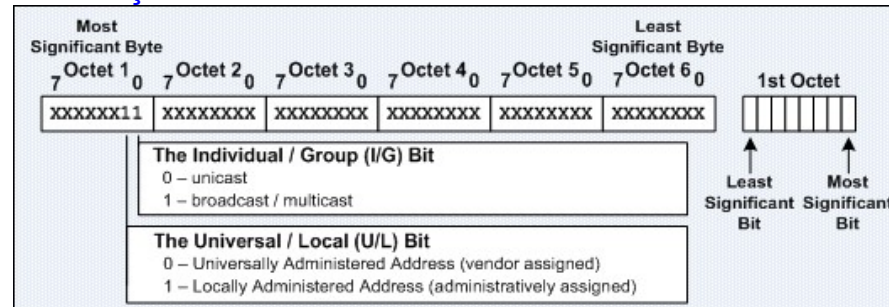


```

v Ethernet II, Src: Cisco_35:f5:b5 (00:10:7b:35:f5:b5), Dst: IPv4mcast_05 (01:00:5e:00:00:05)
  v Destination: IPv4mcast_05 (01:00:5e:00:00:05)
    Address: IPv4mcast_05 (01:00:5e:00:00:05)
    .... 0. .... = LG bit: Globally unique address (factory default)
    .... 1. .... = IG bit: Group address (multicast/broadcast)
  v Source: Cisco_35:f5:b5 (00:10:7b:35:f5:b5)
    Address: Cisco_35:f5:b5 (00:10:7b:35:f5:b5)
    .... 0. .... = LG bit: Globally unique address (factory default)
    .... 0. .... = IG bit: Individual address (unicast)
  Type: IPv4 (0x0800)
> Internet Protocol Version 4, Src: 192.168.23.2, Dst: 224.0.0.5
> Open Shortest Path First
    
```

Ce bit est celui qui type l'adresse et qui indique si c'est un unicast (0 pour Individual) ou un multicast/broadcast, il vaudrait (1 pour Group) dans ce dernier cas. Dans une adresse formulée en IEEE, ce bit serait tout à gauche.

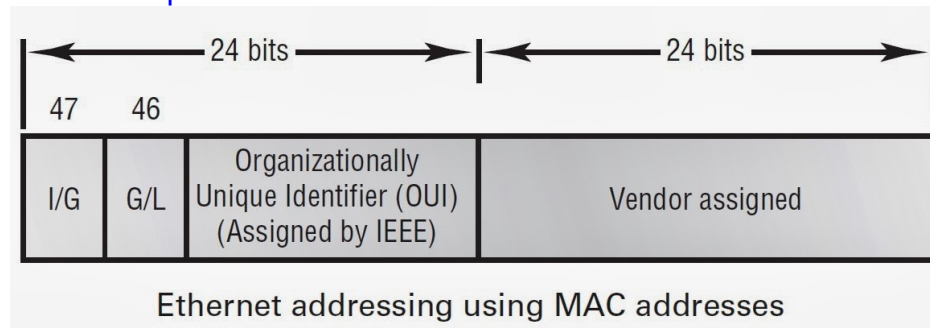
Adresse MAC IEEE802.3 représentée à la façon IETF :



source : <https://www.itcertnotes.com/2012/03/multicast-ip-and-mac-addresses.html> (consultée le 01/12/2019)



Adresse Mac IEEE802.3 au format IEEE tel que dans la norme :



source : <https://learning-network-cisco.blogspot.com/2014/02/> (consulté le 01/12/2019)

Les bits I/G et G/L ne sont pas au même endroit comme on peut le constater.

#### Question 4

Avec quels protocoles de niveau 3 à 4 les paquets OSPF sont acheminés dans le réseau d'après la trace Wireshark ?

##### **Correction :**

Dans la première trace, ça n'est pas visible, il faut voir le cours. Dans le complément donné dans la correction de la question 3, c'est visible, c'est IP. En effet, ici, on est en IPv4 vu les adresses utilisées.

#### Question 5

Dans le contexte OSPF pour l'envoi de messages "Hello Packet", 224.0.0.5 est une adresse IP de destination multicast pourquoi est-elle non routable hors du sous-réseau où elle a été générée ? Par ailleurs son TTL vaut 1 pour quelle raison ?

##### **Correction :**

Avec un TTL de 1, le datagramme contenant le Hello Packet, quand il arrive sur un routeur de son réseau, le voit passer à 0 donc il ne peut aller plus loin.

C'est un Hello Packet qui est fait pour que les routeurs prennent connaissance de leurs voisins sur le même réseau. Par conséquent, ce type de paquet n'a pas vocation à se promener sur Internet

**Question 6**

La trace ci-après doit confirmer votre observation en question 1 sur la périodicité des envois des "Hello Packets", et, vous donner le délai qui indique à partir de quand doit-on considérer qu'un routeur est injoignable (lien en panne ou interface de routeur en panne). Souligner ou surligner cette information dans la trace elle-même.

```

Frame 21: 82 bytes on wire (656 bits), 82 bytes captured (656 bits)
Ethernet II, Src: Cisco_7e:f4:aa (00:10:7b:7e:f4:aa), Dst: IPv4mcast_05 (01:00:5e:00:00:05)
Internet Protocol Version 4, Src: 192.168.23.1, Dst: 224.0.0.5
Open Shortest Path First
  ▾ OSPF Header
    Version: 2
    Message Type: Hello Packet (1)
    Packet Length: 48
    Source OSPF Router: 10.4.4.4
    Area ID: 0.0.0.1
    Checksum: 0x3237 [correct]
    Auth Type: Null (0)
    Auth Data (none): 0000000000000000
  ▾ OSPF Hello Packet
    Network Mask: 255.255.255.0
    Hello Interval [sec]: 10
    > Options: 0x02, (E) External Routing
    Router Priority: 1
    Router Dead Interval [sec]: 40
    Designated Router: 192.168.23.2
    Backup Designated Router: 192.168.23.1
    Active Neighbor: 10.3.3.3

```

**Correction :**

Hello Interval [sec] indique "10". La périodicité des Hello Packet est bien 10 secondes comme nous l'avons observé puis indiqué à la question 1.

Router Dead Interval [sec] indique le temps au bout duquel, sans réception de message, le routeur 192.168.23.2 est considéré comme hors service. La valeur associée est "40", soit 40 secondes avant qu'un routeur qui n'émet plus de Hello Packet soit considéré comme en panne.

