

Plus court chemin

Christophe Viroulaud

Terminale NSI

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Comment fonctionnent les algorithmes de plus court chemin ?

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP : algorithme de Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme de Dijkstra

Principe

Mise en application

Complexité

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

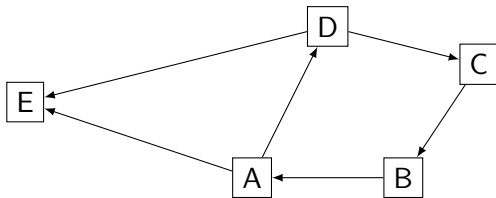


FIGURE – graphe orienté

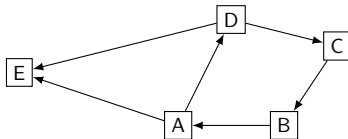


FIGURE – graphe orienté

À retenir

Pour chaque nœud on peut définir ses **prédécesseurs** et ses **successeurs**.

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

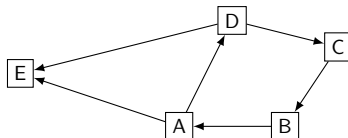
Principe

Mise en application

Complexité

Activité 1 :

1. Établir la matrice d'adjacence du graphe figure 2.
2. Établir le dictionnaire d'adjacence du graphe figure 2.
3. Déterminer un cycle dans le graphe.



Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

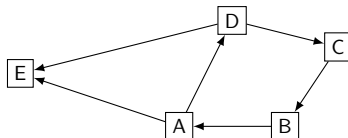
Principe

Mise en application

Complexité

```
1 matrice = [[0, 0, 0, 1, 1],
2             [1, 0, 0, 0, 0],
3             [0, 1, 0, 0, 0],
4             [0, 0, 1, 0, 1],
5             [0, 0, 0, 0, 0]]
```

Code 1 – Matrice d'adjacence



Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

```
1 dico = {"A": {"D", "E"},  
2         "B": {"A"},  
3         "C": {"B"},  
4         "D": {"C", "E"},  
5         "E": set() }
```

Code 2 – Dictionnaire d'adjacence

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

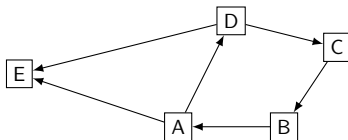
Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité



$A \rightarrow D \rightarrow C \rightarrow B \rightarrow A$

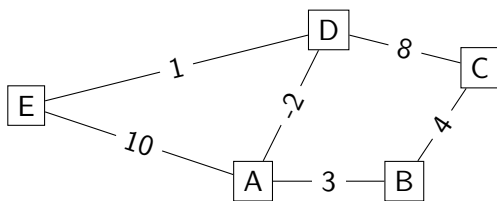


FIGURE – graphe non orienté pondéré

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

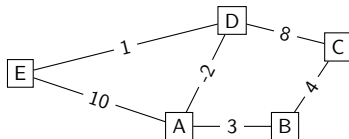
Principe

Mise en application

Complexité

Activité 2 :

1. Établir la matrice d'adjacence du graphe figure 3.
2. Établir le dictionnaire d'adjacence du graphe figure 3.



Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

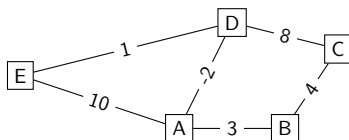
Principe

Mise en application

Complexité

```
1 matrice = [[0, 3, 0, -2, 10],  
2           [3, 0, 4, 0, 0],  
3           [0, 4, 0, 8, 0],  
4           [-2, 0, 8, 0, 1],  
5           [10, 0, 0, 1, 0]]
```

Code 3 – Matrice d'adjacence



Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

```
1 dico = {"A": {"B": 3, "D": -2,  
2         "E": 10},  
3         "B": {"A": 3, "C": 4},  
4         "C": {"B": 4, "D": 8},  
5         "D": {"A": -2, "C": 8, "E":  
              1},  
6         "E": {"D": 1, "E": 10}}
```

Code 4 – Dictionnaire d'adjacence

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

La distance pour atteindre chaque nœud correspond à la distance pour atteindre son prédécesseur à laquelle on ajoute le poids de l'arête les séparant.

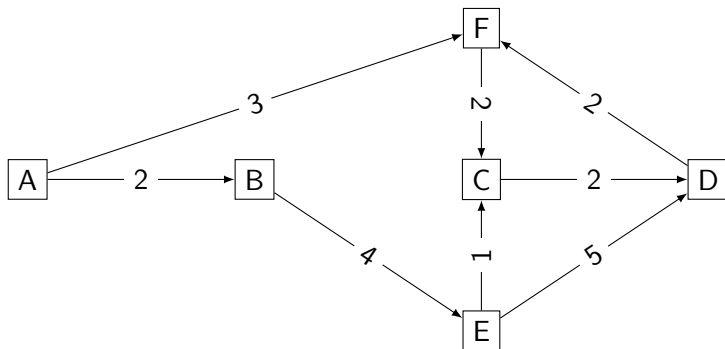


FIGURE – graphe non orienté pondéré

En pratique nous allons utiliser une approche itérative
bottom-up.

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Pour chaque routeur

On obtient un tableau contenant la distance minimale entre le routeur de départ et chaque autre routeur.

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

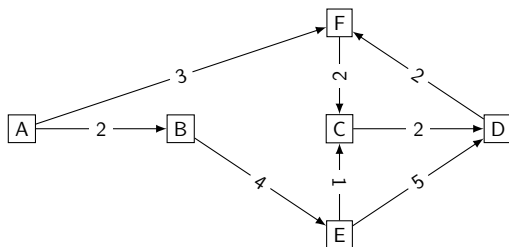
Complexité

```
1 Créer un tableau des distances entre A et les routeurs (A
   inclus), initialisées à l'infini .
2 Dans le tableau modifier la distance vers A à 0.
3
4 Tant que (le nombre d'itérations) < (nombre de routeurs)
5     Pour chaque arc du graphe
6         Si (la distance du routeur) > (la distance de son
           prédécesseur + poids de l'arc entre les deux
           routeurs)
7             La distance du routeur est remplacée par cette
               nouvelle valeur
```

Code 5 – Algorithme de Bellman Ford

Initialisation

Plus court chemin



A	B	C	D	E	F
0	∞	∞	∞	∞	∞

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

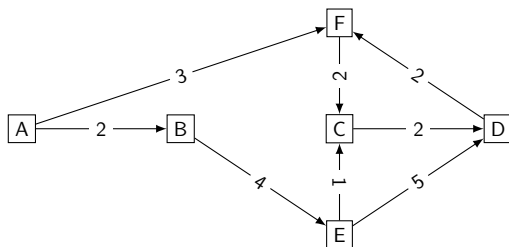
Principe

Mise en application

Complexité

Première itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	∞	∞

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

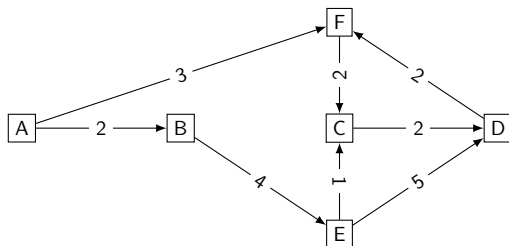
Principe

Mise en application

Complexité

Activité 3 : Continuer de dérouler l'algorithme sur le graphe figure 4.

Première itération



A	B	C	D	E	F
0	2 (A)	∞	∞	∞	∞

Premier prédécesseur de C : E

$$\infty + 1 = \infty \not< \infty$$

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

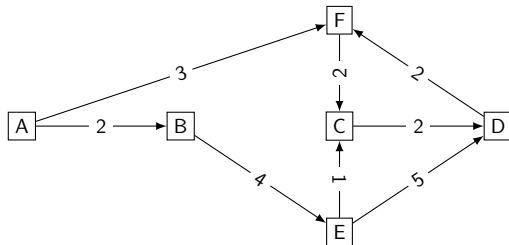
Principe

Mise en application

Complexité

Première itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	∞	∞

Second prédécesseur de C : F

$$\infty + 2 = \infty \neq \infty$$

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

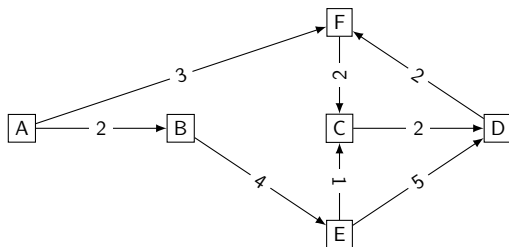
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Première itération



A	B	C	D	E	F
0	2 (A)	∞	∞	∞	∞

Même constat pour D

Plus court chemin

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

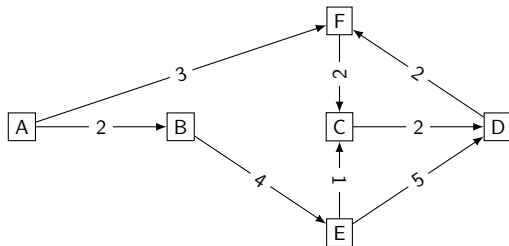
Principe

Mise en application

Complexité

Première itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	∞

Prédécesseur de E : B

$$2 + 4 = 6 < \infty$$

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

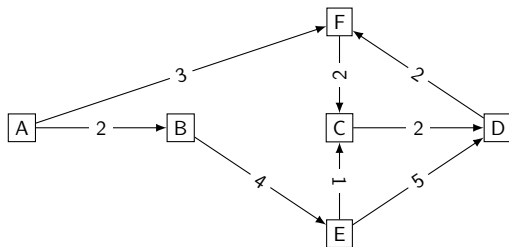
Principe

Mise en application

Complexité

Première itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)

Premier prédécesseur de F : A

$$0 + 3 = 3 < \infty$$

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

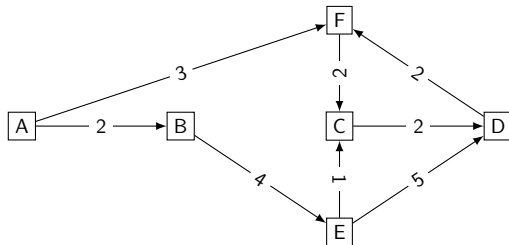
Principe

Mise en application

Complexité

Première itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)

Second prédécesseur de F : D

$$\infty + 2 = \infty \neq \infty$$

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

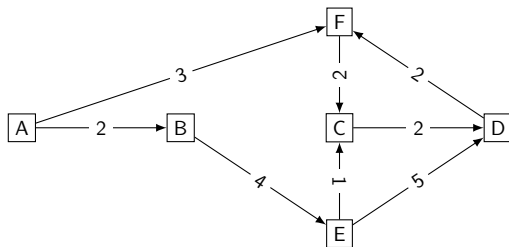
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Première itération



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)

Fin de la première itération

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

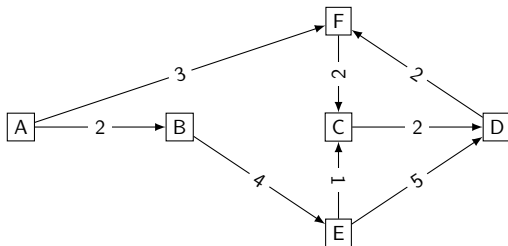
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Deuxième itération



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	∞	∞	6 (B)	3 (A)

Pas de modification pour A et B

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

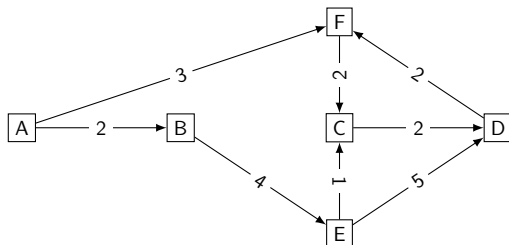
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Deuxième itération

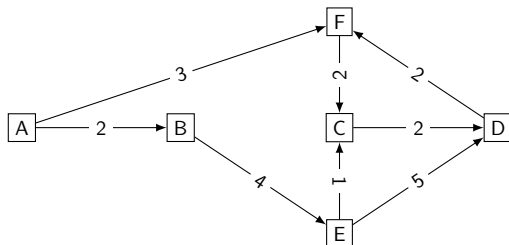


A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	7 (E)	∞	6 (B)	3 (A)

Premier prédécesseur de C : E

$$6 + 1 = 7 < \infty$$

Deuxième itération



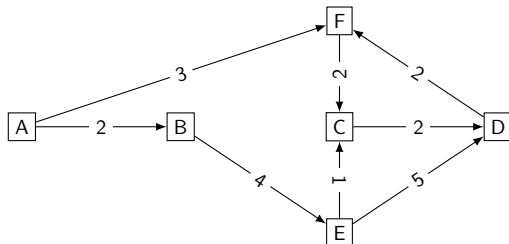
A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	∞	6 (B)	3 (A)

Second prédécesseur de C : F

$$3 + 2 = 5 < 7$$

Deuxième itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)

Premier prédécesseur de D : C

$$5 + 2 = 7 < \infty$$

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

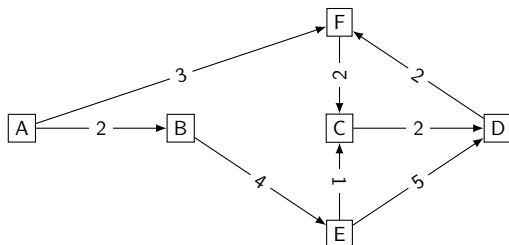
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Deuxième itération

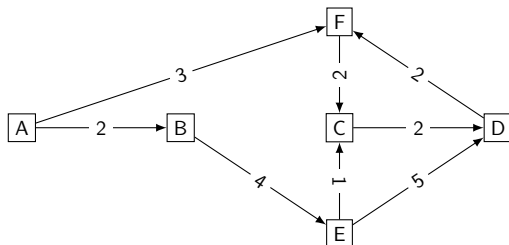


A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)

Second prédécesseur de D : E

$$6 + 5 = 11 \not< 7$$

Deuxième itération



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)

Pas de changement pour E et F

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

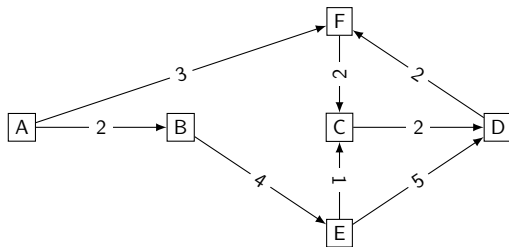
Principe

Mise en application

Complexité

Deuxième itération

Plus court chemin



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)

Fin de la deuxième itération

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :

algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

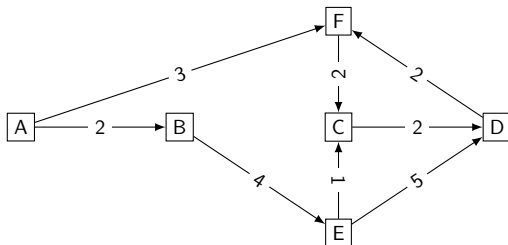
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Troisième itération



A	B	C	D	E	F
0	2 (A)	∞	∞	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)
0	2 (A)	5 (F)	7 (C)	6 (B)	3 (A)

Pas de changement lors de la troisième itération

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

- du nombre de sommets (notée S) : on visite chaque sommet (ligne 4);

```
1  Tant que (le nombre d'itérations)
    < (nombre de routeurs)
```

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

- du nombre de sommets (notée S) : on visite chaque sommet (ligne 4);

```
1 Tant que (le nombre d'itérations)
    < (nombre de routeurs)
```

- du nombre d'arcs (notée A) : pour chaque sommet on regarde tous les arcs du graphe (ligne 5).

```
1 Pour chaque arc du graphe
```

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Complexité de l'algorithme de Bellman Ford

Plus court chemin

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

$$O(S.A)$$

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

principe : construire un sous-graphe en ajoutant à chaque itération un sommet de distance minimale.

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

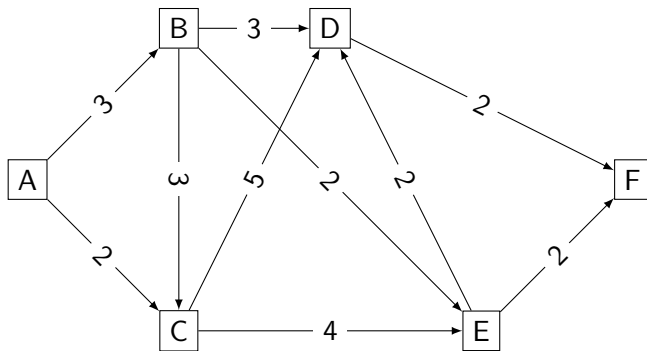


FIGURE – graphe orienté et pondéré

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

1 Créer un tableau des distances entre A et les routeurs
(A inclus), initialisées à l'infini.

2 Dans le tableau modifier la distance vers A à 0.

3
4 Tant qu'il reste des routeurs non sélectionnés

5 Parmi les routeurs non-sélectionnés, choisir le
routeur (noté S) ayant la plus petite distance.

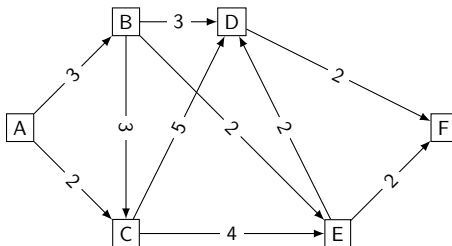
6 Pour chaque routeur adjacent à S (noté V) et non déjà
sélectionné :

7 Si $(\text{la distance de V}) > (\text{la distance de S} +$
poids S-V)

8 La distance de V est remplacée par cette
nouvelle valeur

Code 6 – Algorithme de Dijkstra

Initialisation



A	B	C	D	E	F
0	∞	∞	∞	∞	∞

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

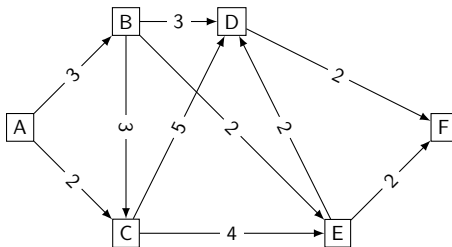
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Sélection de A



A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞

nœuds visités

A

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

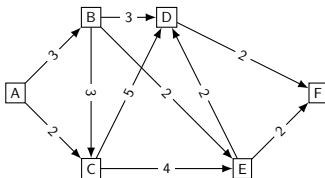
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Activité 4 : Continuer de dérouler l'algorithme sur le graphe figure 5.



On sélectionne le nœud non encore visité et avec la plus petite distance : C.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞

nœuds visités

A - C

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

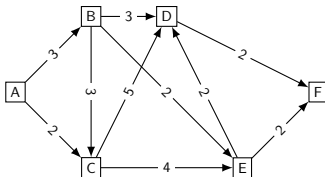
Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité



On sélectionne le nœud non encore visité et avec la plus petite distance : B.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞
	3 (A)		6 (B)	5 (B)	∞

nœuds visités

A - C - B

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

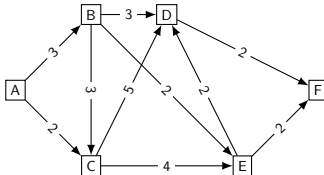
Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité



On sélectionne le nœud non encore visité et avec la plus petite distance : E.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞
	3 (A)		6 (B)	5 (B)	∞
			6 (B)	5 (B)	7 (E)

nœuds visités

A - C - B - E

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

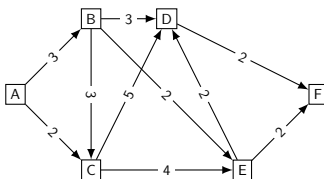
Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité



On sélectionne le nœud non encore visité et avec la plus petite distance : D.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞
	3 (A)		6 (B)	5 (B)	∞
			6 (B)	5 (B)	7 (E)
			6 (B)		7 (E)

nœuds visités

A - C - B - E - D

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

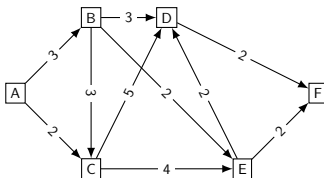
OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Correction



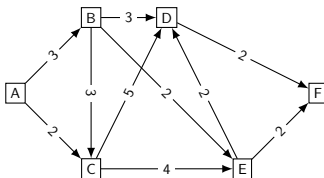
On sélectionne le nœud non encore visité et avec la plus petite distance : F.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞
	3 (A)		6 (B)	5 (B)	∞
			6 (B)	5 (B)	7 (E)
			6 (B)		7 (E)
					7 (E)

nœuds visités

A - C - B - E - D - F

Correction



On sélectionne le nœud non encore visité et avec la plus petite distance : F.

A	B	C	D	E	F
0	3 (A)	2 (A)	∞	∞	∞
	3 (A)	2 (A)	7 (C)	6 (C)	∞
	3 (A)		6 (B)	5 (B)	∞
			6 (B)	5 (B)	7 (E)
			6 (B)		7 (E)
					7 (E)

nœuds visités

A - C - B - E - D - F

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

- La complexité dépend du nombre de sommets S et du nombre d'arcs A .

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

Problématique

Retour des graphes

Graphe orienté

Graphe pondéré

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

- La complexité dépend du nombre de sommets S et du nombre d'arcs A .
- Le point clé de l'algorithme tient dans la recherche de la distance minimale.

- 1 Parmi les routeurs non-sélectionnés, choisir le routeur (noté S) ayant la plus petite distance.

Complexité de l'algorithme de Dijkstra

Plus court chemin

Problématique

Retour des graphes

Graphes orientés

Graphes pondérés

Protocole RIP :
algorithme de
Bellman-Ford

Principe

Mise en application

Complexité

OSPF : algorithme
de Dijkstra

Principe

Mise en application

Complexité

$$O((A + S) \times \log S)$$