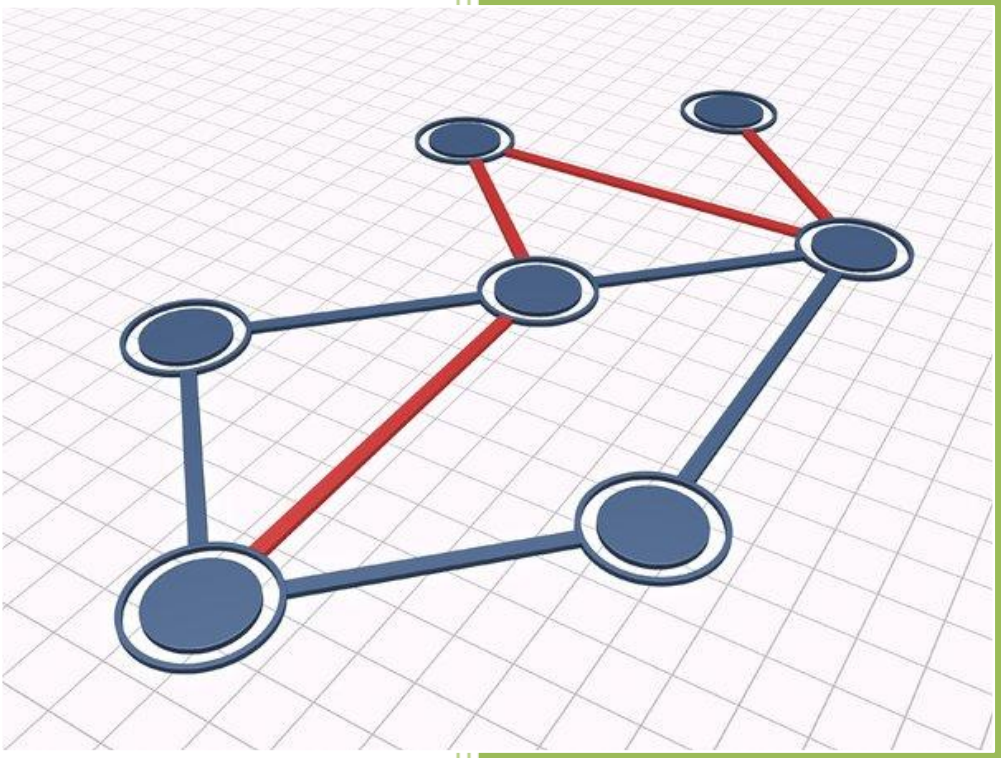
MASTER DS

Projet du module Recherche Opérationnelle

PARTIE 2 (théorie des graphes DIJIKSTRA)



BASTA Mohammed

Cadi Ayyad - FSSM

2017 - 2018

Cette partie concerne l'implémentation de l'algorithme de DIJIKSTRA sous Matlab.

L'une des techniques de définition du chemin le plus court dans la théorie des graphes est DIJKSTRA : Suppose que les poids des arêtes sont des valeurs positives dans la matrice creuse G. La complexité temporelle est O (log (N) * E), où N et E sont respectivement le nombre de nœuds et de bords.

D'abord on va voir la notion de « chemin le plus cours » ensuite la définition de DIJIKSTRA que j'ai utilisé et enfin la démonstration sous MATLAB.

I. INTRODUCTION :

Définitions :

THEORIE DES GRAPHES [th_graphes 2018] :

La théorie des graphes est la discipline mathématique et informatique qui étudie les *graphes*, lesquels sont des modèles abstraits de dessins de réseaux reliant des objets[1]. Ces modèles sont constitués par la donnée de « points », appelés *nœuds* ou *sommets* (en référence aux polyèdres), et de « liens » entre ces points ; ces liens sont souvent symétriques (les graphes sont alors dits *non orientés*) et sont appelés des *arêtes*.

CHEMIN LE PLUS COURS [ch+cours 2018] :

Un problème de plus court chemin est un problème algorithmique de la théorie des graphes. L'objectif est de calculer un chemin entre des sommets d'un graphe qui minimise ou maximise une certaine fonction.

Il existe de nombreuses variantes de ce problème. On suppose donné un graphe fini, orienté ou non selon les cas ; de plus, chaque arc ou arête possède une valeur qui peut être un poids ou une longueur. Un chemin le plus court entre deux nœuds donnés est un chemin qui minimise la somme des valeurs des arcs traversés. Pour calculer un plus court chemin, il existe de nombreux algorithmes, selon la nature des valeurs et d'éventuelles contraintes supplémentaires. Dans de nombreux cas, il existe des algorithmes de complexité en temps polynomiale, comme l'algorithme de Dijkstra dans des graphes avec poids positifs.

ALGORITHME DIJKSTRA [DIJKSTRA 2018] :

l'algorithme de Dijkstra (prononcé [dɛɪkstra]) sert à résoudre le problème du plus court chemin. Il permet, par exemple, de déterminer un plus court chemin pour se rendre d'une ville à une autre connaissant le réseau routier d'une région. Plus précisément, il calcule des plus courts chemins à partir d'une source dans un graphe orienté pondéré par des réels positifs. On peut aussi l'utiliser pour calculer un plus court chemin entre un sommet de départ et un sommet d'arrivée.

Algorithme[dijkstra_algo 2018]**:**

```
Entrées :    un graphe avec une pondération

 positive     des arcs,     un sommet de    pour
chaque sommet



Tant qu'il existe un sommet hors de

    Choisir un sommet     hors de     de plus

petite distance

    Mettre     dans

    Pour chaque sommet     hors de      voisin de



    Fin Pour
Fin Tant Que
```

Fonctions d'aide :

- Assert() : gestion des exceptions
- linprog() : résolution des problèmes linéaire
- numel() : nombre d'éléments dans une matrice
- isnan() : return oui si la variable est numérique
- zeros() : créer un tableau avec des zéros
- plot() : pour présenter les données sur un graphe
- num2str() : convertir numérique en caractère

II. Implémentation :

Documentation :

https://www.mathworks.com/matlabcentral/fileexchange/36140-dijkstra-algorithm

INPUT :

En entrée on lui donne la matrice A :

```
entrez la matrice A:
[0 3 9 0 0 0 0;
0 0 0 7 1 0 0;
0 2 0 7 0 0 0;
0 0 0 0 0 2 8;
0 0 4 5 0 9 0;
0 0 0 0 0 0 4;
0 0 0 0 0 0 0;
];
```

Et le numéro sommet de départ s :

```
entrez le element s:
1
```

Et le numéro du sommet d'arrivé d :

```
entrez le element d:
5
```

OUTPUT :

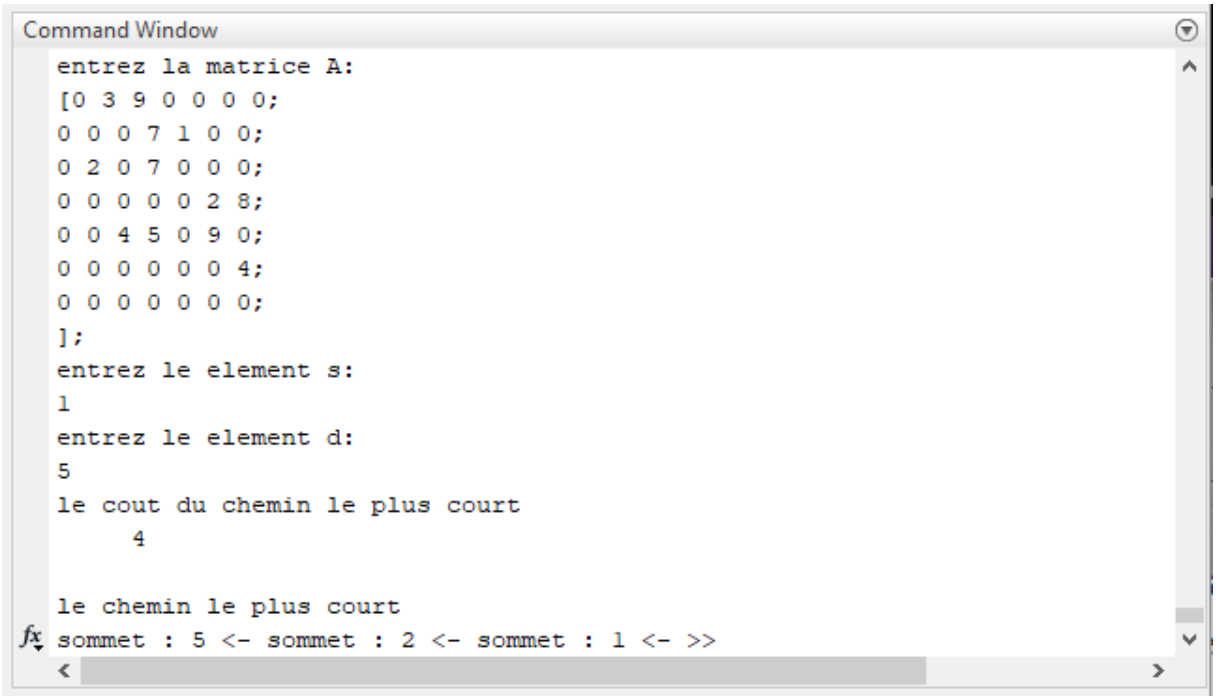
Le cout du chemin optimal :

```
le cout du chemin le plus court
     4
```

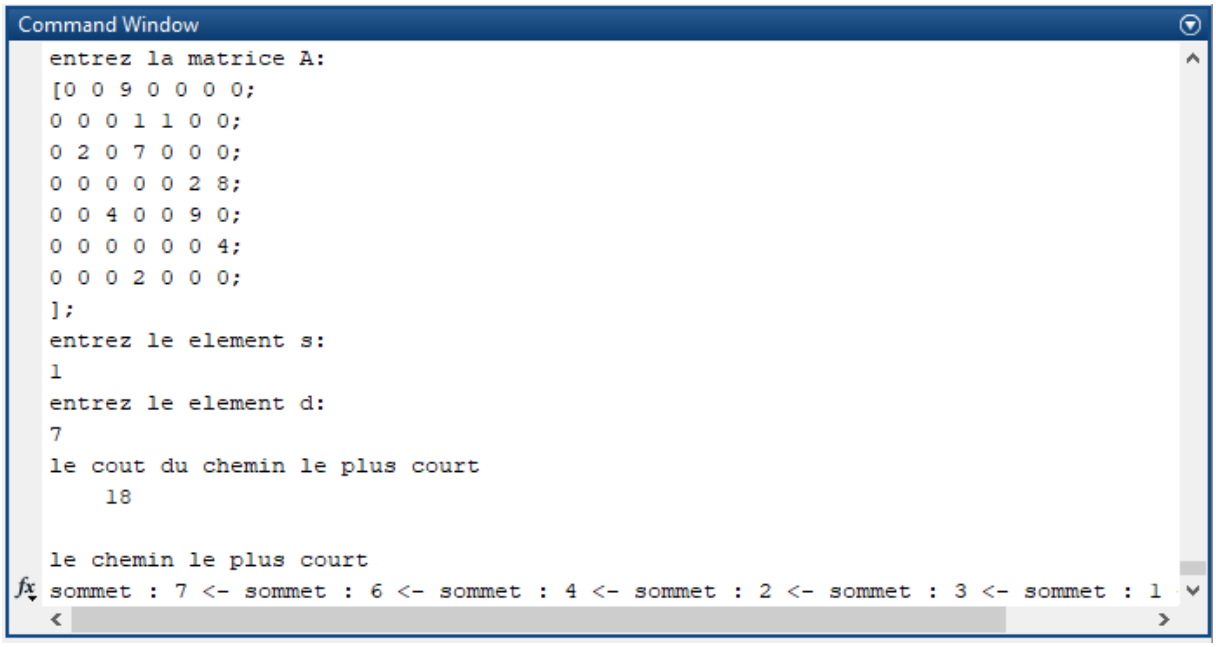
Le chemin optimal :

```
le chemin le plus court
sommet : 5 <- sommet : 2 <- sommet : 1
```

Exemple 1 :



Exemple 2 :



Remarque : ci-joint les fonctions utilisées dans mon implémentation.