Lycee Charlemagne MPSI2 Annee 2021/22

# Dijkstra

Lycee Charlemagne MPSI2 Annee 2021/22

## Edsger

Wikipedia vous dira tout :
Edsger Wybe Dijkstra (prononciation : « dextra ») né à Rotterdam le 11 mai 1930 et mort à Nuenen le 6 août 2002, est un mathématicien et informaticien néerlandais du xxe siècle. Il reçoit en 1972 le prix Turing pour ses contributions sur la science et l'art des langages de programmation et au langage Algol. Juste avant sa mort, en 2002, il reçoit le prix PoDC de l'article influent, pour ses travaux sur l'auto-stabilisation. L'année suivant sa mort, le prix sera renommé en son honneur prix Dijkstra.

Dijkstra avait joué un rôle important dans le développement du langage Algol à la fin des années 1950 et développé ensuite « la science et l'art des langages de programmation », contribuant grandement à la compréhension de leur structure, de leur représentation et de leur implémentation4. C'est aussi un adepte du bel algorithme, y compris pour des sujets difficiles à traiter en programmation structurée comme les perles de Dijkstra (disposer une par une des perles de trois couleurs sur un fil de façon qu'il n'y ait jamais deux séquences adjacentes identiques).
Il est également à l'origine de l'algorithme éponyme, l'algorithme de Dijkstra, permettant de calculer des plus courts chemins dans un graphe orienté. 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 et est très utilisé, par exemple dans les assistants de navigation GPS.

Le discours qu'il prononce en 1972 lorsqu'il reçoit le prix Turing, The Humble Programmer, est également resté célèbre. Il s'agit également d'un exercice d'auto-dérision, le professeur Dijkstra s'étant toujours montré très conscient de la valeur de ses travaux.

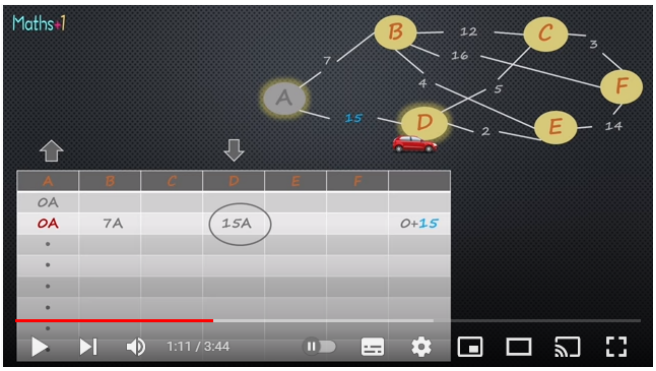
L'aphorisme « L'informatique n'est pas plus la science des ordinateurs que l'astronomie n'est celle des télescopes. » souvent attribué à Dijkstra, est en fait une phrase de Michael R. Fellows et Ian Parberry dans un article du journal Computing Research News.

Lycee Charlemagne MPSI2 Annee 2021/22

## Algorithme

Yvan Monka explique mieux que très bien :
https ://www.youtube.com/watch ?v=rHylCtXtdNs

Et ensuite, en « cinq minutes pour comprendre » :
https ://www.youtube.com/watch ?v=MybdP4kice4



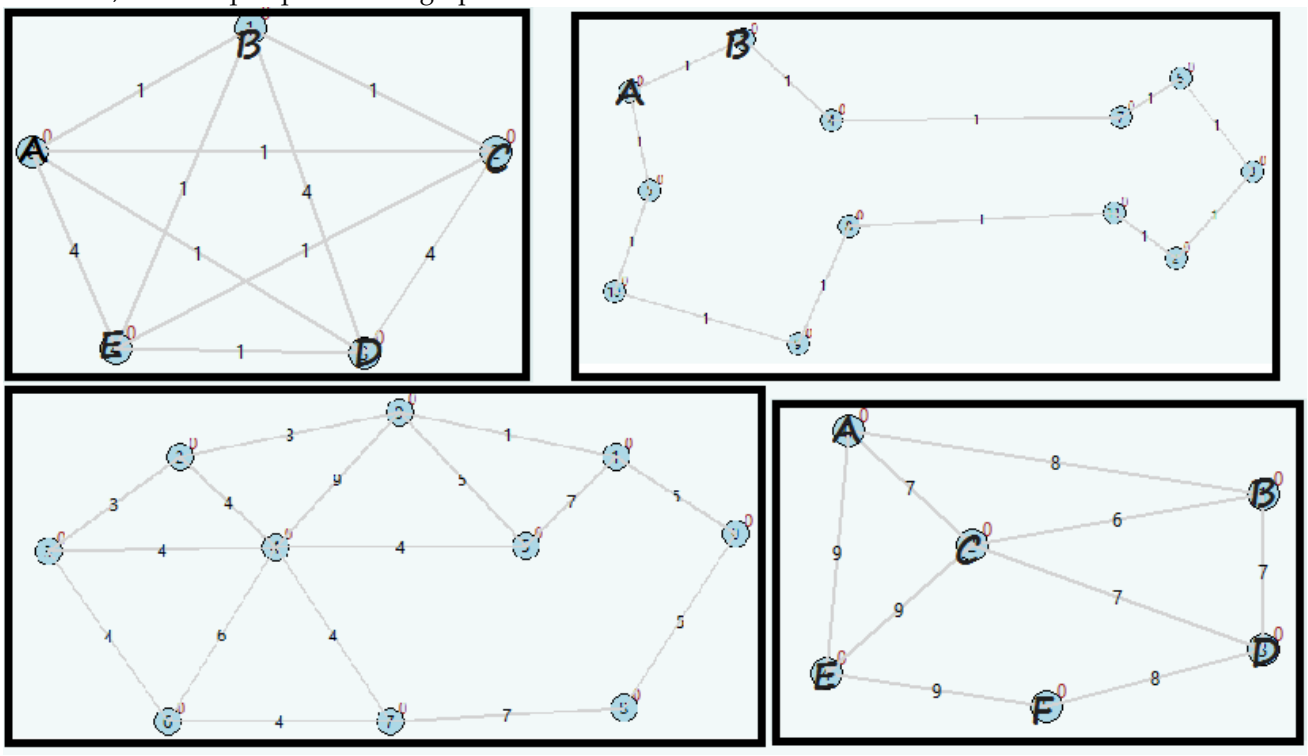

Allez, trouvez le plus court chemin avant la fin de la vidéo...

| A | B | C | D | E | F | |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Lycee Charlemagne | MPSI2 | Annee 2021/22

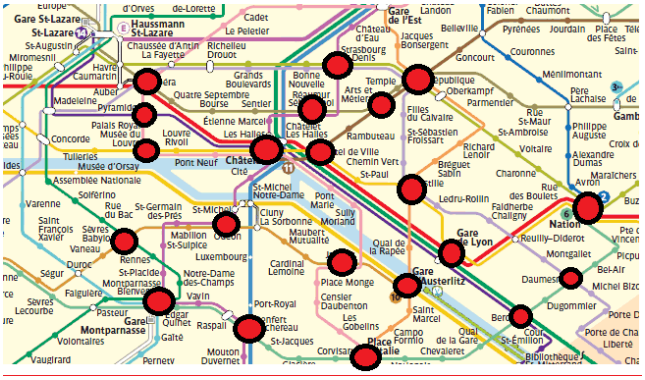
## Utilitaire graphe

J'ai conçu pour vous un utilitaire Python pour créer des graphes et leur matrice. A charger sur Discord ou sur cahier de Prépas.
Lancez le, et créez quelques uns de graphes ci dessous :




Appliquez à chacun l'algorithme de Dijkstra en cliquant sur la touche dédiée. Si nécessaire, modifiez des paramètres pour suivre l'exécution pas à pas.

Ou créez même vos propres graphes. Ou celui d'Yvan Monka.
Et lancez le bouton « Dijkstra ».



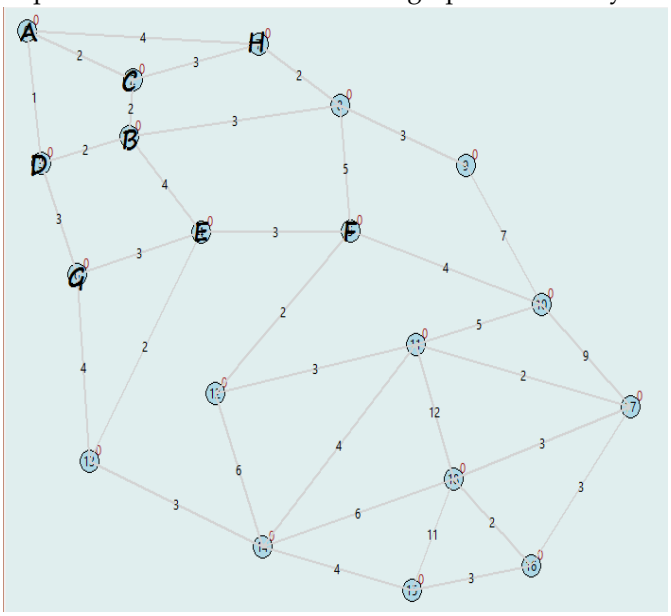

Prenez le graphe du métro pour le centre de Paris. Les sommets eront les grosses stations avec correspondances. Les arêtes seront les lignes entre deux stations, et le poids d'une arête sera le nombre de stations entre les deux sommets.

Appliquez l'algorithme de Dijkstra.

Regardez ce qu'il faut changer au programme pour pouvoir choisir la station de départ.

---

Reproduisez à l'aide de l'utilitaire graphe codé en Python le graphe suivant :




| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | 2 | 1 | | | | 4 | | | | | | | | | | | |
| B | | | 2 | 2 | 4 | | | | 3 | | | | | | | | | | |
| C | 2 | 2 | | | | | | 3 | | | | | | | | | | | |
| D | 1 | 2 | | | | | 3 | | | | | | | | | | | | |
| E | | 4 | | | | 3 | 3 | | | | | | | | 2 | | | | |
| F | | | | | 3 | | | | 5 | | 4 | | 2 | | | | | | |
| G | | | | 3 | 3 | | | | | | | | | | 4 | | | | |
| H | 4 | | 3 | | | | | | | | | | | | | | | | |
| I | | 3 | | | | 5 | | 2 | | | | | | | | | | | |
| J | | | | | | | | | | | | | | | | | | | |
| K | | | | | | 4 | | | | | | 5 | | | | | | 9 | |
| L | | | | | | | | | | | | | 3 | | 4 | | | 2 | 12 |
| M | | | | | | 2 | | | | | | 3 | | | 6 | | | | |
| N | | | | | 2 | | 4 | | | | | | | | 3 | | | | |
| O | | | | | | | | | | | | 4 | 6 | 3 | | 4 | | | 6 |
| P | | | | | | | | | | | | | | | 4 | | 3 | | 11 |
| O | | | | | | | | | | | | | | | | 3 | | 3 | 2 |
| R | | | | | | | | | | | | 9 | 2 | | | | 3 | | 3 |
| S | | | | | | | | | | | | | 12 | | 6 | 11 | 2 | 3 | |

Les numéros des villes ne sont pas très lisibles ? Aidez vous de la matrice...

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | O | R | S |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | |

Appliquez l'algorithme de Dijkstra.
Quelle est la ville la plus éloignée de la ville 0 ?

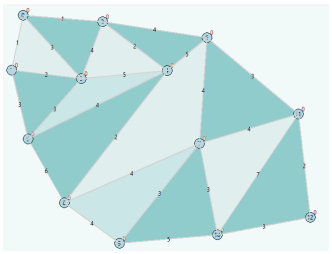
| Lycee Charlemagne | MPSI2 | Annee 2021/22 |
|---|---|---|
| . | Table | . |

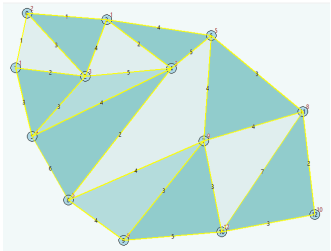
La table est donnée, tracez le graphe, il en existe une version où les routes ne se croisent pas.

| | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | 1 | 3 | 1 | | | | | | | | | |
| B | 1 | | 2 | | | 3 | | | | | | | |
| C | 3 | 2 | | 4 | 5 | 3 | | | | | | | |
| D | 1 | | 4 | | 2 | | 4 | | | | | | |
| E | | | 5 | 2 | | 4 | 5 | | 2 | | | | |
| F | | 3 | 3 | | 4 | | | | 6 | | | | |
| G | | | | 4 | 5 | | | 4 | | | | 3 | |
| H | | | | | | 4 | | 4 | 3 | 3 | 4 | | |
| I | | | | | 2 | 6 | | 4 | | 4 | | | |
| J | | | | | | | | 3 | 4 | | 5 | | |
| K | | | | | | | | 3 | | 5 | | 7 | 3 |
| L | | | | | | | 3 | 4 | | | 7 | | 2 |
| M | | | | | | | | | | | 3 | 2 | |

| A | B | C | D | E | F | G | H | I | J | K | L | M | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |
| | | | | | | | | | | | | | |

Vérification :



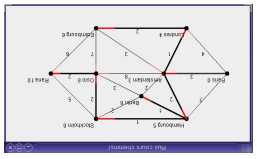


Lycee Charlemagne MPSI2 Annee 2021/22

## Villes

Avec des noms de villes.
Voici la réponse finale. Retrouvez le graphe (unique ?) dont c'est la solution.
Les chemins et les distance depuis Paris sont :

Pour Paris : Paris, distance 0
Pour Hambourg : Paris Amsterdam Hambourg, distance 5
Pour Amsterdam : Paris Amsterdam, distance 3
Pour Londres : Paris Amsterdam Londres, distance 4
Pour Stockholm : Paris Amsterdam Hambourg Stockholm, distance 6
Pour Berlin : Paris Amsterdam Hambourg Berlin, distance 6
Pour Édimbourg : Paris Amsterdam Londres Édimbourg, distance 6
Pour Oslo : Paris Amsterdam Hambourg Stockholm Oslo, distance 8
Pour Riga : Paris Amsterdam Hambourg Stockholm Oslo Riga, distance 10



Lycee Charlemagne MPSI2 Annee 2021/22

## Programmation

Pseudo-code :

```
Entrées : G=(S, poids) un graphe avec une pondération positive
          poids des arcs,
          s un sommet de S
Q := [tous les sommets]
d[a] :=+∞ pour chaque sommet a (sauf s)
d[s] = 0
predecesseur[s] = aucun

tant que Q est non vide
....choisir un sommet a de Q de plus petite distance d[a] [a]
....enlever a de Q
....pour chaque sommet b de Q voisin de a
........si d[b] > d[a] + poids(a, b) [b]
............d[b] = d[a]+poids(a,b)
........prédécesseur[b] := a
....fin pour
fin tant que
```

*a.* qui sera ce sommet a à la première étape ?
*b.* quel est l'intérêt de l'initialisation « d[a] :=+∞ pour chaque sommet a sauf s » ?

```python
def initialisation (s, ens_sommets) :
    D = [[inf, nan]for k in range (len(ens_sommets))]
    D[s] = [0, s]
    return D

def trouve_min (Q, D):
    mini = inf
    sommet = -1
    for s in Q :
        d = D [s][0]
        if d < mini :
            mini = d
            sommet = s
    return sommet

def maj_distances (G, D, s1, s2):
    if D[s2][0] > D[s1][0] + G[s1][s2] :
        D[s2][0] = D[s1][0] + G[s1][s2]
        D[s2][1] = s1

def Dijkstra (G, s, ens_sommets):
    D= initialisation (s, ens_sommets)
    Q = [elem for elem in G]
    while Q != [] :
        s1 = trouve_min (Q, D)
        if s1 == -1 :
            return D
        Q.remove (s1)
        for s2 in G[s1]:
            maj_distances (G, D, s1, s2)
    return D
```

1 - C'est quoi la constante `inf` dans `initialisation` et `trouve_mini` ?

-

2 - C'est quoi la constante `nan` dans `initialisation` ?

-

3 - A quoi correspond la réponse `-1` dans `trouve_min`.

-

4 - Sous quelle forme doit être donne `G` dans `Dijkstra` ?

-

5 - A quoi sert `Q = [elem for elem in G]` ?

-

6 - Qui est `s` parmi les trois variables entrées dans `Dijkstra` ?

-

7 - Précisez les spécification des fonctions (type de variables en entrée, type de réponse).

-

8

- Justifiez que `Dijkstra` termine (et si possible, trouvez sa complexité).

-

Terminaison : pourquoi l'algorithme n'effectue qu'un nombre fini d'étapes ?
A chaque étape, $Q$ perd un élément.

Correction : pourquoi l'algorithme cherche bien le plus court chemin.
On pose l'invariant de boucle : "à chaque itération de la boucle while/tant que, pour tout $s$ hors de $Q$ , $d[v] = dist(s, v)$ ».
en appelant $dist$ la fonction « longueur du plus court chemin entre les deux sommets » [1]
On montre par récurrence cet invariant.

Initialisation
Comme $Q$ = tous, l'invariant est vrai.

Conservation
Montrons qu'à chaque itération "$d[b] = dist(s, b)$ pour le sommet $b$ enlevé à $Q$ ».
Par l'absurde, supposons par l'absurde que $b$ est le premier sommet pour lequel $d[b]$ soit différent $dist(s, b)$ quand on l'enlève à $Q$.

$d$ n'est pas $s$ car $d[s] = 0 = dist(s, s)$.

Il existe un chemin de $s$ à $b$ sinon $d[b] = +\infty = d(s, b)$ .
Il existe donc un plus court chemin de $s$ à $b$ qu'on va appeler $p$.
On peut alors décomposer p de la manière suivante $[s, \ldots, x, y, \ldots, b]$ (les points de suspension peuvent être vides) où $y$ est le premier sommet appartenant à $Q$ et $x$ son prédécesseur (ce peut être $s$).
On a $d[y] = dist(s, y)$ au moment où on enlève$b$ de $Q$.
Comme $x$ n'est pas dans $Q$, alors $d[x] = dist(s, x)$.
Dans ce cas, l'arc $(x, y)$ est relâché à cet instant.
Donc $d[y] = dist(s, y)$.
On souhaite faire apparaitre la contradiction suivante : $dist(s, y) \leqslant dist(s, u)$.
$d[y] = dist(s, y) \leqslant dist(s, u) \leqslant d[u]$ (car $y$ arrive avant $u$ sur le chemin le plus court et $dist$ est minimale). Mais comme $b$, $y$ est dans $Q$ lors du choix de $b$, $d[b] \leqslant d[y]$ (car choisi par $trouve_min$).
Donc, on a l'égalité (contradiction).

D'où la correction.

1. son existence est assurée car on travaille sur des ensembles finis, notion de plus petit élément