

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.

Définition

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.
- Il en pose les bases en résolvant le problème des 7 ponts de Königsberg en 1740.

Définition

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.
- Il en pose les bases en résolvant le problème des 7 ponts de Königsberg en 1740.
- Les graphes interviennent à présent dans de nombreux problèmes (recherche de chemins, réseau, ...) en informatique comme en mathématiques.

Définition

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.
- Il en pose les bases en résolvant le problème des 7 ponts de Königsberg en 1740.
- Les graphes interviennent à présent dans de nombreux problèmes (recherche de chemins, réseau, ...) en informatique comme en mathématiques.

Définition

Un **graphe orienté** est la donnée :

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.
- Il en pose les bases en résolvant le problème des 7 ponts de Königsberg en 1740.
- Les graphes interviennent à présent dans de nombreux problèmes (recherche de chemins, réseau, ...) en informatique comme en mathématiques.

Définition

Un **graphe orienté** est la donnée :

- D'un ensemble de **sommets** S (V pour *vertex* en anglais.).

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Aspect historique

- C'est le mathématicien suisse Leonhard Euler (1707-1783) qui est à l'origine de la création de la théorie des graphes.
- Il en pose les bases en résolvant le problème des 7 ponts de Königsberg en 1740.
- Les graphes interviennent à présent dans de nombreux problèmes (recherche de chemins, réseau, ...) en informatique comme en mathématiques.

Définition

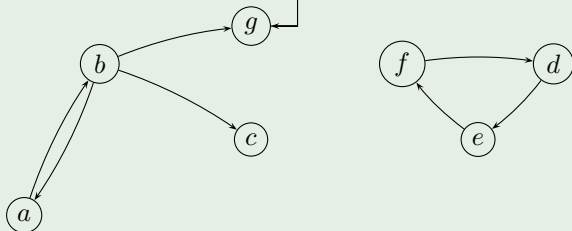
Un **graphe orienté** est la donnée :

- D'un ensemble de **sommets** S (V pour *vertex* en anglais.).
- D'un ensemble de couples de sommets $A \subseteq S \times S$ appelés **arc** (notés $x \rightarrow y$). (E pour *edges* en anglais).

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

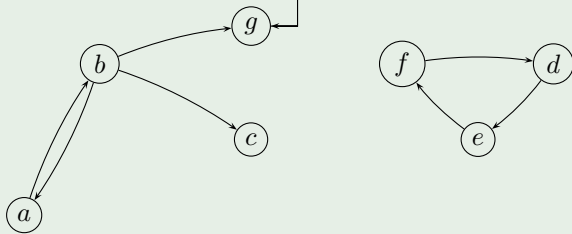
Exemple



C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

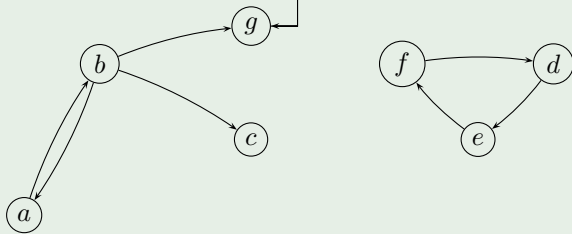


$$S = \{a, b, c, d, e, f, g\}$$

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple



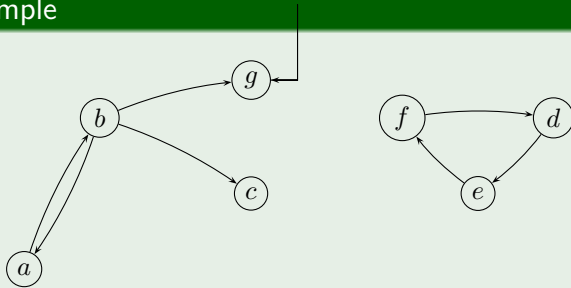
$$S = \{a, b, c, d, e, f, g\}$$

$$A = \{(e, f), (d, e), (f, d), (a, b), (b, a), (b, c), (g, b)\}$$

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple



$$S = \{a, b, c, d, e, f, g\}$$

$$A = \{(e, f), (d, e), (f, d), (a, b), (b, a), (b, c), (g, b)\}$$

⚠ Seule la données de S et A défini le graphe et *pas* les positions des sommets sur le schéma.

Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.
- Un arc de la forme (x, x) est une **boucle**.
- Le **degré sortant** d'un sommet s noté $d_+(s)$ est le nombre d'arcs de la forme (s, x) .

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.
- Un arc de la forme (x, x) est une **boucle**.
- Le **degré sortant** d'un sommet s noté $d_+(s)$ est le nombre d'arcs de la forme (s, x) .
- Le **degré entrant** d'un sommet s noté $d_-(s)$ est le nombre d'arcs de la forme (x, s) .

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.
- Un arc de la forme (x, x) est une **boucle**.
- Le **degré sortant** d'un sommet s noté $d_+(s)$ est le nombre d'arcs de la forme (s, x) .
- Le **degré entrant** d'un sommet s noté $d_-(s)$ est le nombre d'arcs de la forme (x, s) .
- Un **chemin** de longueur n du sommet s au sommet t dans un graphe (S, A) est une séquence x_0, \dots, x_n de sommets tels que $x_0 = s$, $x_n = t$ et $(x_i, x_{i+1}) \in A$ pour $i \in \llbracket 0; n-1 \rrbracket$.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.
- Un arc de la forme (x, x) est une **boucle**.
- Le **degré sortant** d'un sommet s noté $d_+(s)$ est le nombre d'arcs de la forme (s, x) .
- Le **degré entrant** d'un sommet s noté $d_-(s)$ est le nombre d'arcs de la forme (x, s) .
- Un **chemin** de longueur n du sommet s au sommet t dans un graphe (S, A) est une séquence x_0, \dots, x_n de sommets tels que $x_0 = s$, $x_n = t$ et $(x_i, x_{i+1}) \in A$ pour $i \in \llbracket 0; n-1 \rrbracket$.
- Un chemin est **simple** lorsqu'il est sans répétition d'arcs.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

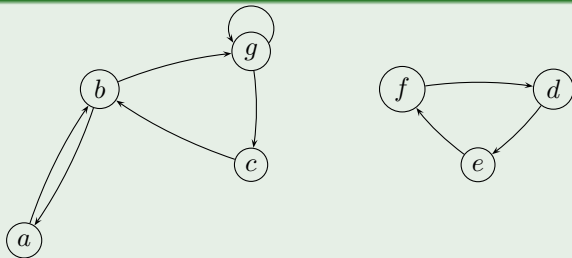
Vocabulaire

- Le **degré** d'un graphe est son nombre de sommets.
- Un arc de la forme (x, x) est une **boucle**.
- Le **degré sortant** d'un sommet s noté $d_+(s)$ est le nombre d'arcs de la forme (s, x) .
- Le **degré entrant** d'un sommet s noté $d_-(s)$ est le nombre d'arcs de la forme (x, s) .
- Un **chemin** de longueur n du sommet s au sommet t dans un graphe (S, A) est une séquence x_0, \dots, x_n de sommets tels que $x_0 = s$, $x_n = t$ et $(x_i, x_{i+1}) \in A$ pour $i \in \llbracket 0; n-1 \rrbracket$.
- Un chemin est **simple** lorsqu'il est sans répétition d'arcs.
- Un **cycle** est un chemin simple d'un sommet à lui même.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

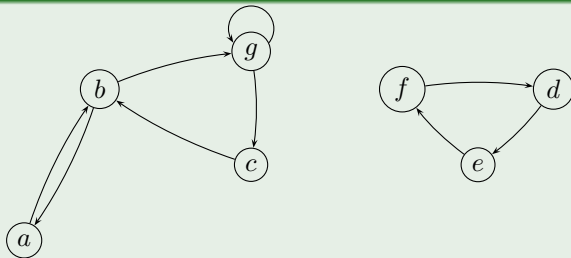


- Donner $d_+(b)$.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

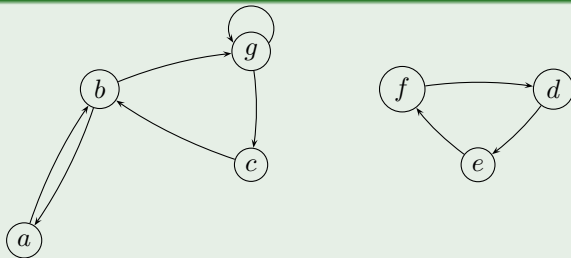


- Donner $d_+(b)$.
- Donner $d_-(g)$.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple



- Donner $d_+(b)$.
- Donner $d_-(g)$.
- Donner un chemin sans cycle de q à a .

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.
- Les définitions de chemin, degrés, ... des graphes orientés s'étendent naturellement aux graphes non orientés.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.
- Les définitions de chemin, degrés, ... des graphes orientés s'étendent naturellement aux graphes non orientés.
- On dit qu'un graphe non orienté (S, A) est **connexe** lorsqu'il existe un chemin entre toute paire de sommets.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Un **graphe non orienté** est la donnée :

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.
- Les définitions de chemin, degrés, ... des graphes orientés s'étendent naturellement aux graphes non orientés.
- On dit qu'un graphe non orienté (S, A) est **connexe** lorsqu'il existe un chemin entre toute paire de sommets.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Un **graphe non orienté** est la donnée :

- D'un ensemble de **sommets ou noeuds** S .

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.
- Les définitions de chemin, degrés, ... des graphes orientés s'étendent naturellement aux graphes non orientés.
- On dit qu'un graphe non orienté (S, A) est **connexe** lorsqu'il existe un chemin entre toute paire de sommets.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Définition

Un **graphe non orienté** est la donnée :

- D'un ensemble de **sommets ou noeuds** S .
- D'un ensemble de **paires** de sommets A appelés **arcs ou arêtes** notés $x - y$.

Vocabulaire

- Dans le contexte des graphes orientés cela revient à $(x, y) \in A$ ssi $(y, x) \in A$.
- Les définitions de chemin, degrés, ... des graphes orientés s'étendent naturellement aux graphes non orientés.
- On dit qu'un graphe non orienté (S, A) est **connexe** lorsqu'il existe un chemin entre toute paire de sommets.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Graphes pondérés

- Dans de nombreuses situations, on est amené à attacher une information aux arcs d'un graphe (ex : distance entre deux villes, coût d'une liaison dans un réseau informatique, ...), on parle alors de **graphe pondéré**.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Graphes pondérés

- Dans de nombreuses situations, on est amené à attacher une information aux arcs d'un graphe (ex : distance entre deux villes, coût d'une liaison dans un réseau informatique, ...), on parle alors de **graphe pondéré**.
- L'information, ou **étiquette** attaché à un noeud est souvent de nature numérique, on parle alors de **poids** ou **longueur** d'un arc.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Graphes pondérés

- Dans de nombreuses situations, on est amené à attacher une information aux arcs d'un graphe (ex : distance entre deux villes, coût d'une liaison dans un réseau informatique, ...), on parle alors de **graphe pondéré**.
- L'information, ou **étiquette** attaché à un noeud est souvent de nature numérique, on parle alors de **poids** ou **longueur** d'un arc.
- Le coût d'un chemin est alors la somme des poids des arcs qui le compose.

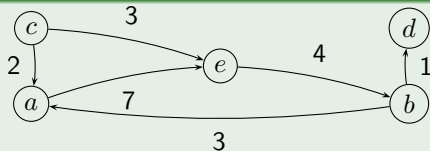
C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Graphes pondérés

- Dans de nombreuses situations, on est amené à attacher une information aux arcs d'un graphe (ex : distance entre deux villes, coût d'une liaison dans un réseau informatique, ...), on parle alors de **graphe pondéré**.
- L'information, ou **étiquette** attaché à un noeud est souvent de nature numérique, on parle alors de **poids** ou **longueur** d'un arc.
- Le coût d'un chemin est alors la somme des poids des arcs qui le compose.

Exemple



C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

- On numérote les sommets du graphe

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

- On numérote les sommets du graphe
- S'il y a une arête du sommet i vers le sommet j alors on place un 1 à la ligne i et à la colonne j de M

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

- On numérote les sommets du graphe
- S'il y a une arête du sommet i vers le sommet j alors on place un 1 à la ligne i et à la colonne j de M
- Sinon on place un 0

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

- On numérote les sommets du graphe
- S'il y a une arête du sommet i vers le sommet j alors on place un 1 à la ligne i et à la colonne j de M
- Sinon on place un 0

Remarques

- Si le graphe n'est pas orienté alors la matrice est symétrique par rapport à sa première diagonale.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par matrice d'adjacence

On peut représenter un graphe à n sommets par sa **matrice d'adjacence** M , c'est à dire un tableau de n lignes et n colonnes :

- On numérote les sommets du graphe
- S'il y a une arête du sommet i vers le sommet j alors on place un 1 à la ligne i et à la colonne j de M
- Sinon on place un 0

Remarques

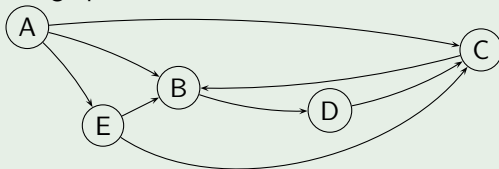
- Si le graphe n'est pas orienté alors la matrice est symétrique par rapport à sa première diagonale.
- On peut représenter les graphes pondérés en écrivant le poids à la place du 1 pour chaque arête.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

- ① En supposant les sommets numérotés dans l'ordre alphabétique, écrire la matrice d'adjacence du graphe suivant :

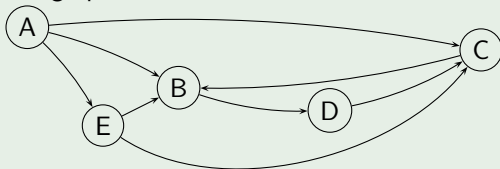


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

- ① En supposant les sommets numérotés dans l'ordre alphabétique, écrire la matrice d'adjacence du graphe suivant :



- ② Dessiner le graphe ayant la matrice d'adjacence suivante (on appellera les sommets S_1, S_2, \dots) :

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

- On crée pour chaque sommet du graphe une liste

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

- On crée pour chaque sommet du graphe une liste
- S'il y a une arête du sommet S_i vers le sommet S_j alors S_j est dans la liste de S_i

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

- On crée pour chaque sommet du graphe une liste
- S'il y a une arête du sommet S_i vers le sommet S_j alors S_j est dans la liste de S_i

Remarques

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

- On crée pour chaque sommet du graphe une liste
- S'il y a une arête du sommet S_i vers le sommet S_j alors S_j est dans la liste de S_i

Remarques

- Lorsqu'un graphe a "peu" d'arête cette implémentation est plus intéressante en terme d'occupation mémoire que celle par matrice d'adjacence.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Représentation par listes d'adjacence

On peut représenter un graphe à l'aide de listes d'adjacences, c'est à dire en mémorisant pour chaque sommet du graphe la liste de ses voisins.

- On crée pour chaque sommet du graphe une liste
- S'il y a une arête du sommet S_i vers le sommet S_j alors S_j est dans la liste de S_i

Remarques

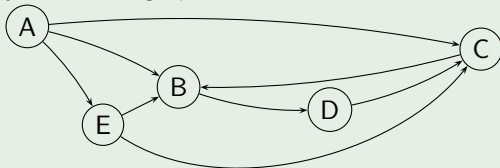
- Lorsqu'un graphe a "peu" d'arête cette implémentation est plus intéressante en terme d'occupation mémoire que celle par matrice d'adjacence.
- En Python, on utilisera un dictionnaire pour représenter les listes d'adjacences, les clés sont les sommets et les valeurs les listes associées

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

- 1 Ecrire les listes d'adjacences du graphe suivante :

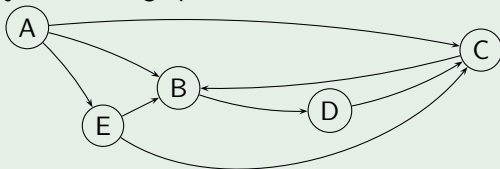


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple

- ① Ecrire les listes d'adjacences du graphe suivante :



- ② Dessiner le graphe représenté par le dictionnaire Python suivante :

```
{  
    'A' : ['C'],  
    'B' : ['D', 'E'],  
    'C' : ['A', 'B'],  
    'D' : ['A', 'C'],  
    'E' : ['B', 'C', 'D']  
}
```

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Parcours d'un graphe

A la base des algorithmes sur les graphes, on trouve les parcours de graphe, c'est à dire l'exploration des sommets. A partir du sommet de départ, on peut :

- explorer tous ses voisins immédiats, puis les voisins des voisins et ainsi de suite. Le graphe est donc exploré en « cercle concentrique » autour du sommet de départ . . . , on parle alors de **parcours en largeur** ou **breadth first search** (*BFS*) en anglais.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Parcours d'un graphe

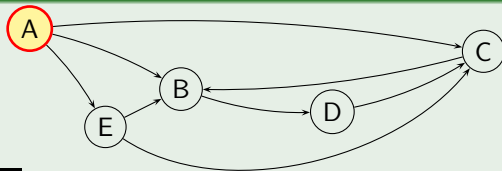
A la base des algorithmes sur les graphes, on trouve les parcours de graphe, c'est à dire l'exploration des sommets. A partir du sommet de départ, on peut :

- explorer tous ses voisins immédiats, puis les voisins des voisins et ainsi de suite. Le graphe est donc exploré en « cercle concentrique » autour du sommet de départ . . . , on parle alors de **parcours en largeur** ou **breadth first search** (*BFS*) en anglais.
- explorer à chaque étape le premier voisin non encore exploré. Lorsque qu'on atteint un sommet dont tous les voisins ont déjà été exploré, on revient en arrière, on parle alors de **parcours en profondeur** ou **depth first search** (*DFS*) en anglais.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en largeur

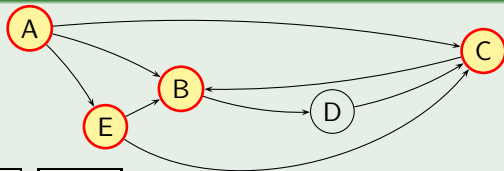


Sommets explorés : A

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en largeur

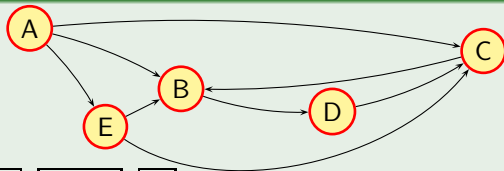


Sommets explorés : A, B,C,E

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en largeur

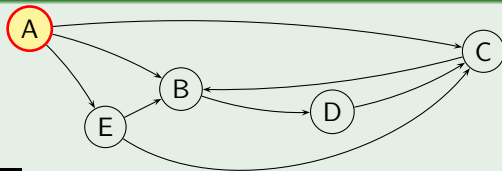


Sommets explorés : A, B,C,E, D.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en profondeur

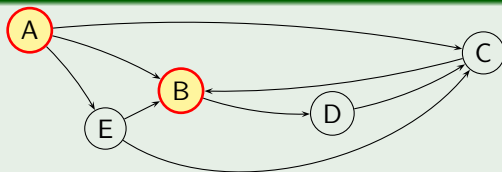


Sommets explorés : A

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en profondeur

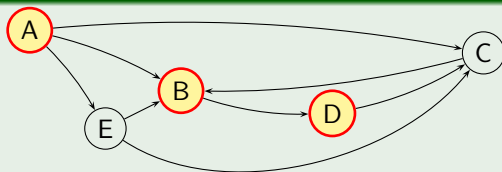


Sommets explorés : A, B

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en profondeur

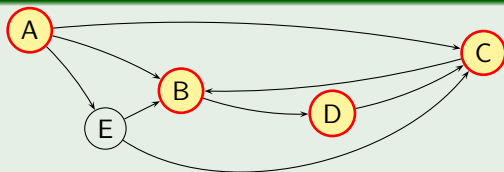


Sommets explorés : A, B, D

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en profondeur

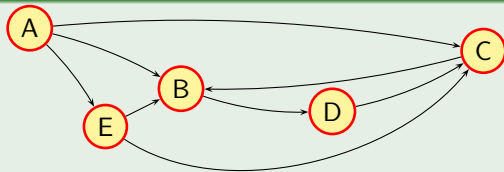


Sommets explorés : A, B, D, C

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Exemple de parcours en profondeur



Sommets explorés : A, B, D, C, E

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins . . . Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins . . . Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins . . . Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.



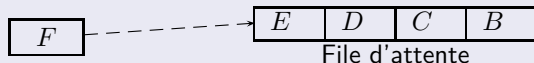
File d'attente

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.

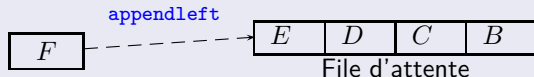


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.

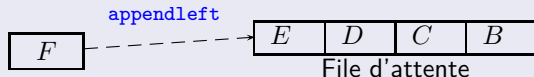


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.

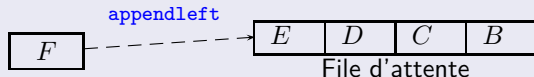


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.

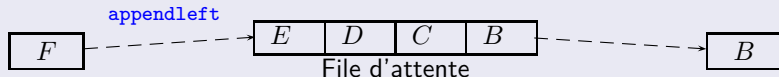


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.

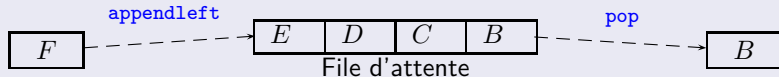


C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en largeur

- Pour un parcours en largeur, on doit stocker dans une structure de données les sommets en attente d'être explorés. C'est à dire les voisins du sommet de départ, puis les voisins des voisins ... Ces sommets doivent être retirés pour exploration, dans leur ordre d'insertion, la structure de données utilisée est donc du type **premier entré, premier sorti** (first in first out (*FIFO*)).
- Ce type de structure de données s'appelle une **file**.
- Pour l'implémentation on doit pouvoir **enfiler** (ajouter un sommet dans la file) et **défiler** (retirer un sommet) de façon efficace donc en $O(1)$.
- Les listes de Python ne sont pas adaptées, on utilisera le module **deque** de Python, enfiler correspond alors à **appendleft** et défiler à **pop**.



C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en profondeur

- Pour un parcours en profondeur, on stocker aussi dans une structure de données les sommets en attente d'être explorés. Mais cette fois, la structure de données utilisée est donc du type **dernier entré, premier sorti** (last in first out (*LIFO*)).

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

File et parcours en profondeur

- Pour un parcours en profondeur, on stocker aussi dans une structure de données les sommets en attente d'être explorés. Mais cette fois, la structure de données utilisée est donc du type **dernier entré, premier sorti** (last in first out (*LIFO*)).
- Ce type de structure de données s'appelle une **pile**.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

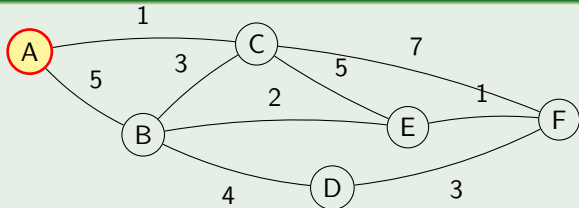
File et parcours en profondeur

- Pour un parcours en profondeur, on stocker aussi dans une structure de données les sommets en attente d'être explorés. Mais cette fois, la structure de données utilisée est donc du type **dernier entré, premier sorti** (last in first out (*LIFO*)).
- Ce type de structure de données s'appelle une **pile**.
- Pour l'implémentation, on se contente d'utiliser la récursivité de façon à ce que la pile des sommets en attente d'être exploré soit gérée de façon automatique par les appels récursifs.

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

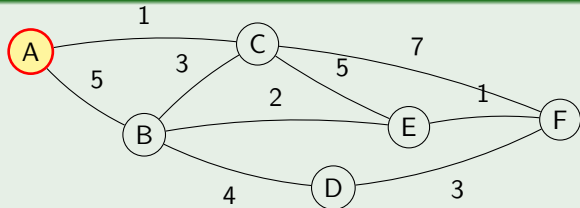


A	B	C	D	E	F	

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

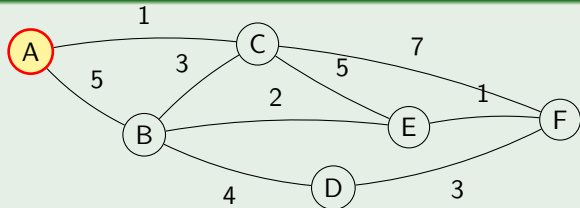


A	B	C	D	E	F	
0 (A)						

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

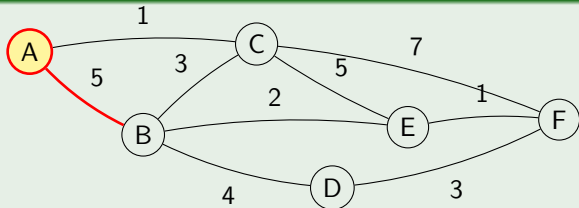


A	B	C	D	E	F	
0 (A)						A
✓						
✓						
✓						
✓						
✓						

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

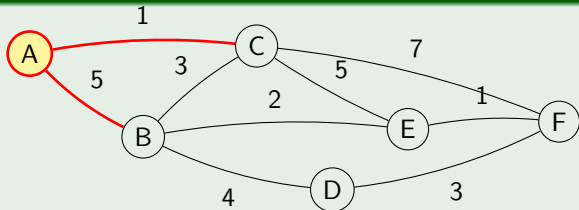


A	B	C	D	E	F	
0 (A)	5 (A)					A
✓						
✓						
✓						
✓						
✓						

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

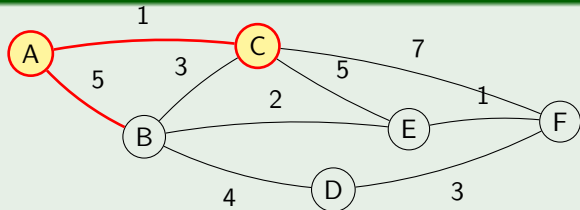


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓						
✓						
✓						
✓						
✓						

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

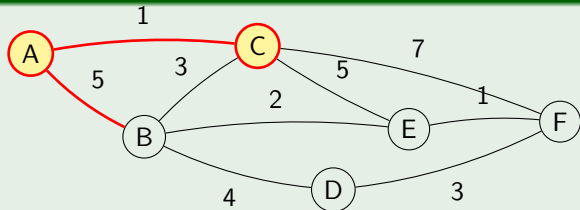


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓		1 (A)				
✓						
✓						
✓						
✓						

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

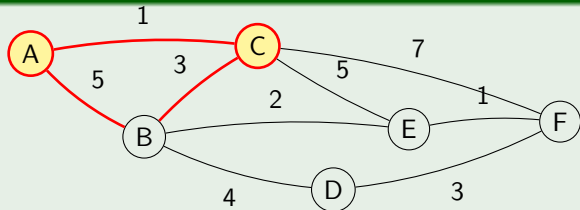


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓		1 (A)				C
✓		✓				
✓		✓				
✓		✓				
✓		✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

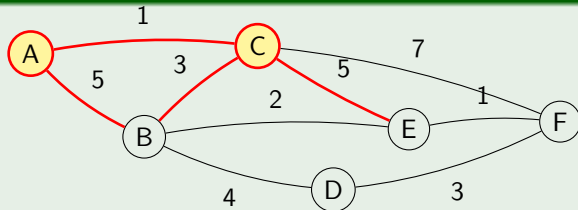


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)				C
✓		✓				
✓		✓				
✓		✓				
✓		✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

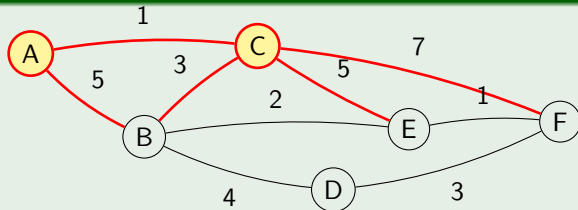


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)		C
✓		✓				
✓		✓				
✓		✓				
✓		✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

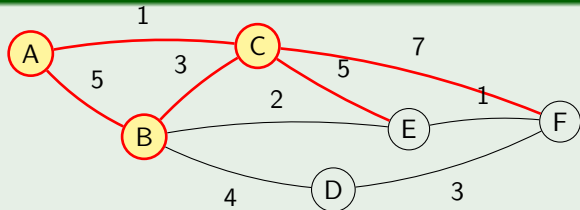


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓		✓				
✓		✓				
✓		✓				
✓		✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

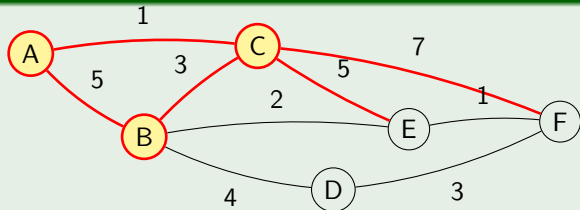


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓				
✓		✓				
✓		✓				
✓		✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

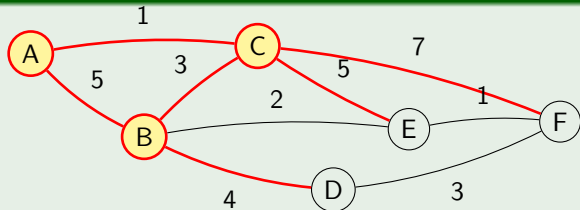


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓				B
✓	✓	✓				
✓	✓	✓				
✓	✓	✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

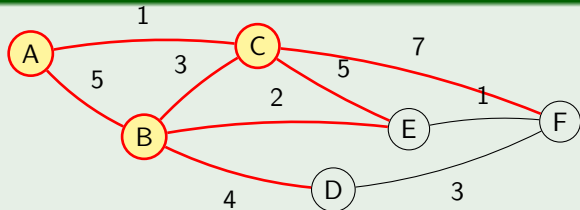


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)			B
✓	✓	✓				
✓	✓	✓				
✓	✓	✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

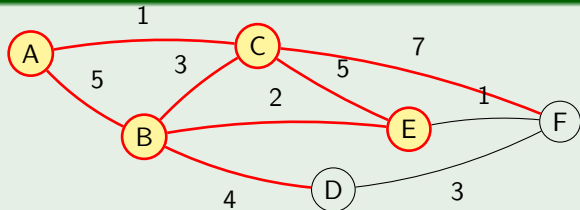


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓				
✓	✓	✓				
✓	✓	✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

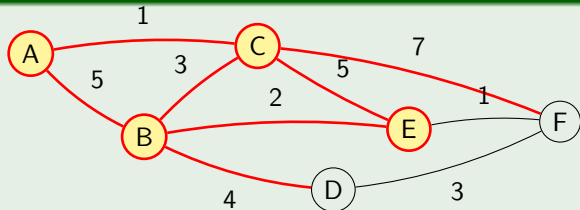


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)		
✓	✓	✓				
✓	✓	✓				

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

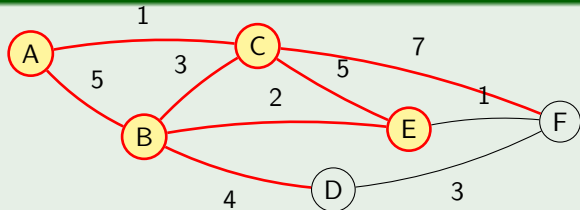


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)		E
✓	✓	✓		✓		
✓	✓	✓		✓		

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

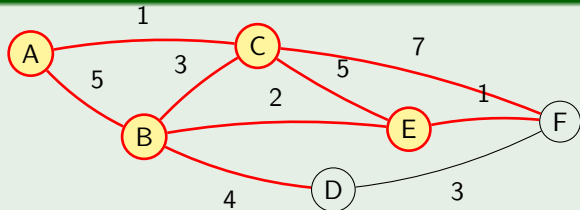


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)		E
✓	✓	✓		✓		
✓	✓	✓		✓		

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

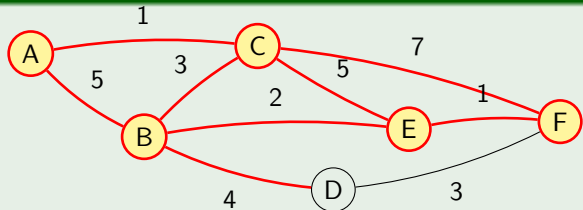


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓		✓		
✓	✓	✓		✓		

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

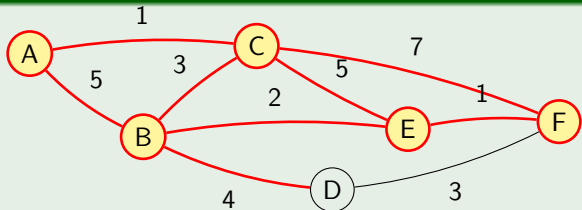


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓		✓	7 (E)	
✓	✓	✓		✓	✓	

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

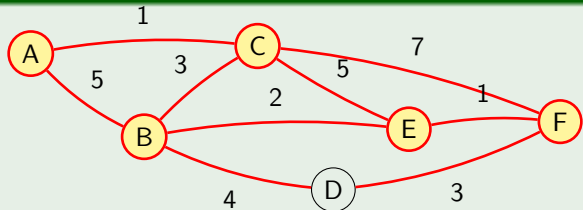


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓		✓	7 (E)	F
✓	✓	✓		✓	✓	

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

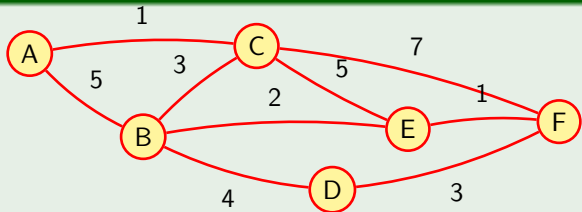


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓	11 (F)	✓	7 (E)	F
✓	✓	✓		✓	✓	

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple

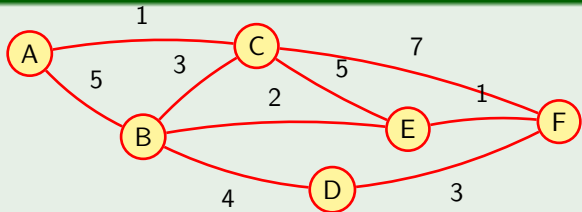


A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓	11 (F)	✓	7 (E)	F
✓	✓	✓	8 (B)	✓	✓	

C4 Algorithmique pour l'IA et l'étude des jeux

1. Rappels sur les graphes

Algorithme de Dijkstra : exemple



A	B	C	D	E	F	
0 (A)	5 (A)	1 (A)				A
✓	4 (C)	1 (A)		6 (C)	8 (C)	C
✓	4 (C)	✓	8 (B)	6 (B)		B
✓	✓	✓		6 (B)	7 (E)	E
✓	✓	✓	11 (F)	✓	7 (E)	F
✓	✓	✓	8 (B)	✓	✓	D