Graphes

Quelques exercices

Mathilde Vernet
mathilde.vernet@univ-lehavre.fr

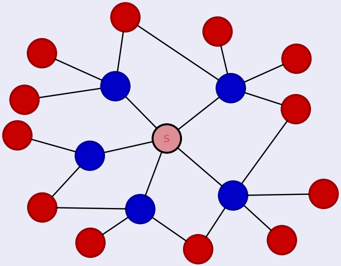
Master Informatique, Master Mathématiques
Université Le Havre Normandie

Automne 2020

Question

Trouver un algorithme permettant de déterminer si un graphe est biparti.

Idée



- Parcourir le graphe
- Colorer les sommets
- Parcourir chaque arête pour vérifier que ses extrémités sont de couleurs différentes

OU

- Parcourir le graphe
- Colorer un sommet en s'assurant que ses voisins ne sont pas de la même couleur

Procedure `estBiparti`(G)
  v.visited $\leftarrow$ false $\forall v \in V$
  s.visited $\leftarrow$ true, s.color $\leftarrow$ rouge, open.`add`(s)
  TantQue $\neg$open.`empty()` **Faire**
    $v \leftarrow$ open.`peek()`
    $u \leftarrow$ un voisin de v avec u.visited $=$ false
    Si u existe **Alors**
      u.visited $\leftarrow$ true
      Si v.color $=$ rouge **Alors** u.color $\leftarrow$ bleu **Sinon** u.color $\leftarrow$ rouge
      open.`add`(u)
    Sinon
      open.`remove()`
    FinSi
  FinTantQue
  biparti=true
  Pour $(u, v) \in E$ **ET TantQue** biparti=true **Faire**
    biparti $\leftarrow$ u.color $\neq$ v.color
  FinPour
  Retourner biparti
FinProcedure

Remarque

On suppose le graphe connexe, s'il ne l'est pas, on recommencera cette procédure sur chaque composante connexe.

Correction

- Lors du parcours en largeur, on colore tous les voisins du sommet courant par la couleur opposé à la sienne, donc on obtient bien la bipartition des sommets si le graphe est biparti.
- Pour chaque arête, on vérifie si les extrémités ont des couleurs différentes, donc le booléen `biparti` indique bien si le graphe est biparti.

Complexité

- Un BFS : $O(n+m)$
- Une itération sur toutes les arêtes : $O(m)$

TOTAL : $O(n+m)$

Procedure `estBiparti`(G)

  v.visited $\leftarrow$ false $\forall v \in V$

  s.visited $\leftarrow$ true, s.color $\leftarrow$ rouge, open.`add`(s)

  biparti=true

  TantQue $\neg$open.`empty()` **ET** biparti=true **Faire**

    $v \leftarrow$ open.`peek()`

    PourTout $(v, u) \in E$ **ET** biparti=true **Faire**

      Si u.visited=true **Alors**

        biparti $\leftarrow$ u.color $\neq$ v.color

      Sinon

        u.visited $\leftarrow$ true

        Si v.color $=$ rouge **Alors** u.color $\leftarrow$ bleu **Sinon** u.color $\leftarrow$ rouge

        open.`add`(u)

      FinSi

    FinPour

    open.`remove()`

  FinTantQue

  Retourner biparti

FinProcedure

Remarque

On suppose le graphe connexe, s'il ne l'est pas, on recommencera cette procédure sur chaque composante connexe.

Correction

- Lors du parcours en largeur, on colore tous les voisins du sommet courant par la couleur opposé à la sienne, donc on obtient bien la bipartition des sommets si le graphe est biparti.
- Quand deux sommets voisins sont déjà colorés, on vérifie si les extrémités ont des couleurs différentes, donc le booléen `biparti` indique bien si le graphe est biparti.

Complexité

- Un BFS : $O(n+m)$
- Pendant le BFS, on ne fait que des opérations en temps constant

TOTAL : $O(n+m)$

Exercice

Je suis dans un labyrinthe et je dois en sortir. Il y a différents couloirs et des intersections dans ce labyrinthe. Le couloir d'une intersection à une autre coute un certain nombre de pièces d'or. Certains couloirs entre intersections sont particuliers puisque si je les emprunte alors je ne paye pas mais je gagne des pièces d'or. Les couloirs ne peuvent être empruntés que dans un sens sous peine de terribles représailles du gardien du labyrinthe. Par chance, je dispose de la liste de tous les couloirs, avec le cout ou le gain de les emprunter.

- De l'entrée vers l'intersection A coute 2 pièces d'or
- De l'entrée vers l'intersection B coute 2 pièces d'or
- De l'intersection B vers l'intersection A offre 1 pièce d'or
- De l'intersection A vers l'intersection C offre 3 pièces d'or
- De l'intersection A vers l'intersection D offre 3 pièces d'or
- De l'intersection A vers la sortie coute 1 pièce d'or
- De l'intersection C vers l'intersection B coute 5 pièces d'or
- De l'intersection C vers l'intersection D est gratuit
- De l'intersection C vers la sortie coute 2 pièces d'or
- De l'intersection D vers la sortie coute 3 pièces d'or

Puisque j'ai suivi le cours de graphes, je vais pouvoir sortir du labyrinthe en dépensant le moins possible. Comment faire ? Détaillez votre réflexion et chacun de vos choix.

Comment représenter ce problème sous forme de graphe ?

- L'entrée du labyrinthe, sa sortie, ainsi que chaque intersection entre des couloirs sont des sommets du graphe
- Un arête entre deux sommets représente le fait que le couloir correspondant existe
- Un poids sur une arête représente le cout en pièces d'or du couloir correspondant. Pour les couloirs qui font gagner des pièces d'or, alors on mettra un poids négatif égal au gain de ce couloir.

Quel est le problème ?

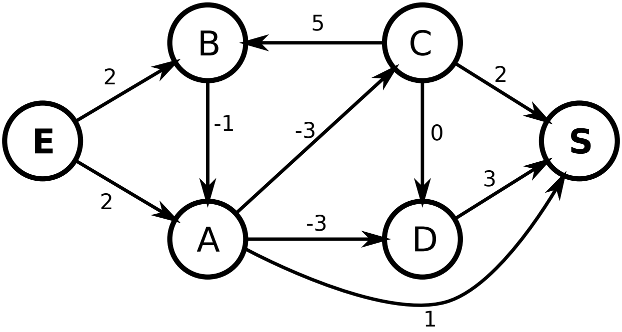
- Je veux aller de l'entrée à la sortie du labyrinthe en perdant le moins de pièces d'or possible
- Je cherche donc un plus court chemin depuis l'entrée du labyrinthe dans le graphe que j'ai décrit

Quels sont les algorithmes envisageable ?

- Je cherche un plus court chemin depuis un sommet : Floyd-Warshall n'est donc pas adapté
- Le graphe a des cycles : l'algorithme pour les DAG n'est donc pas adapté
- Le graphe a des poids négatifs : Dijkstra n'est donc pas adapté
- J'utilise Bellman-Ford

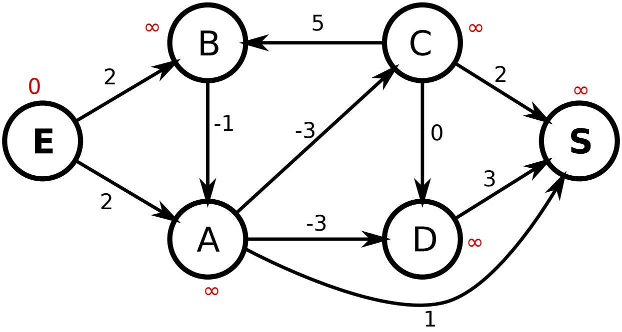
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



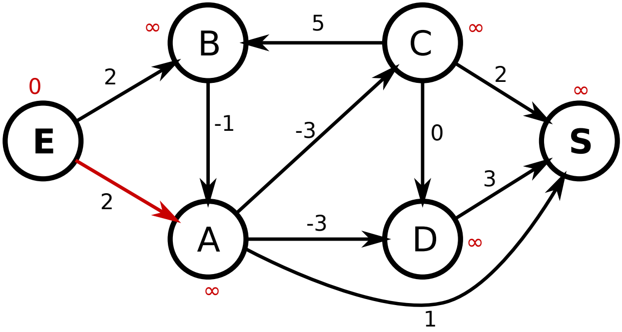
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



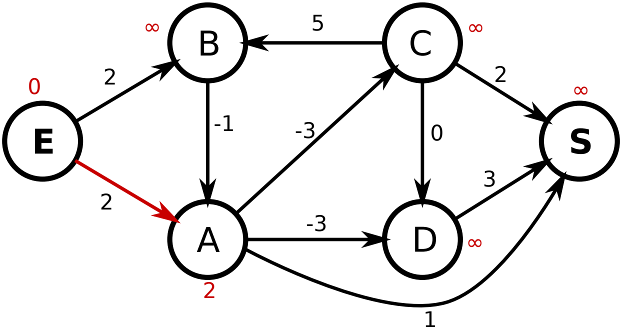
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



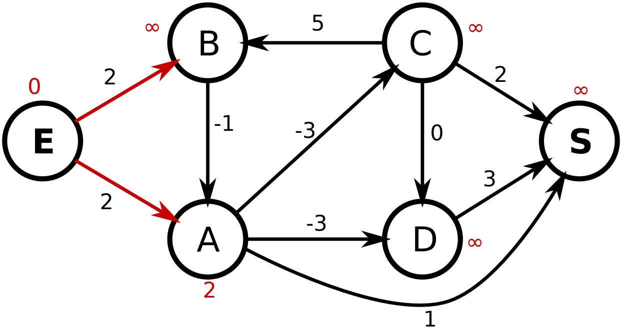
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



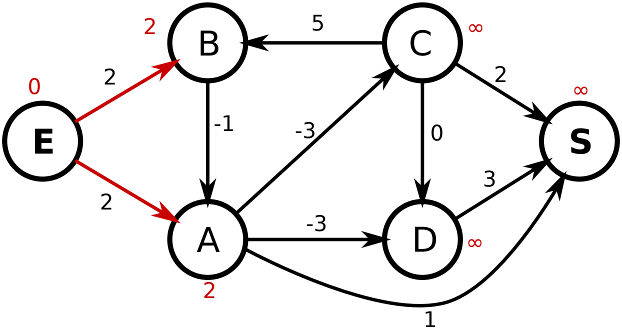
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



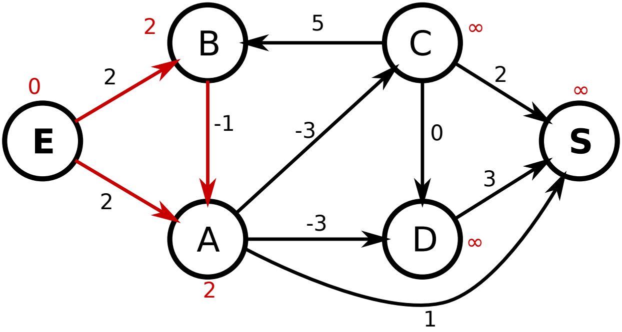
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



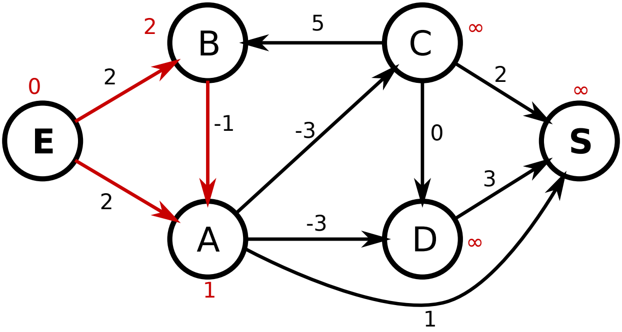
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



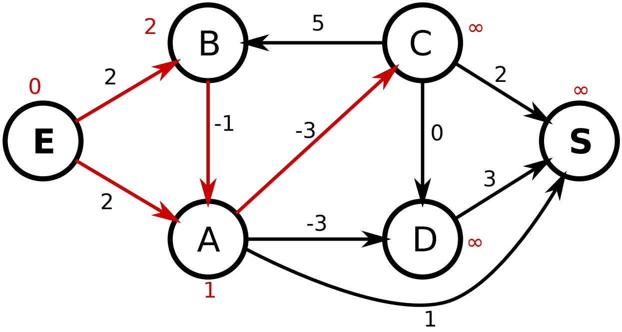
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



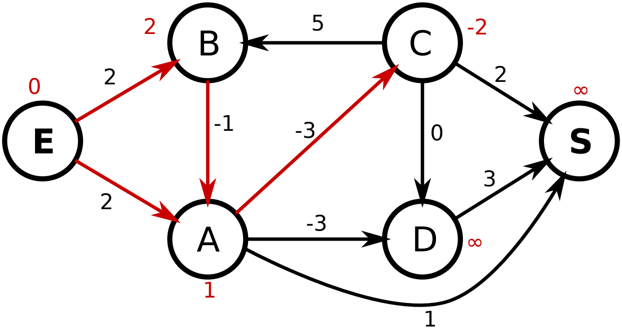
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



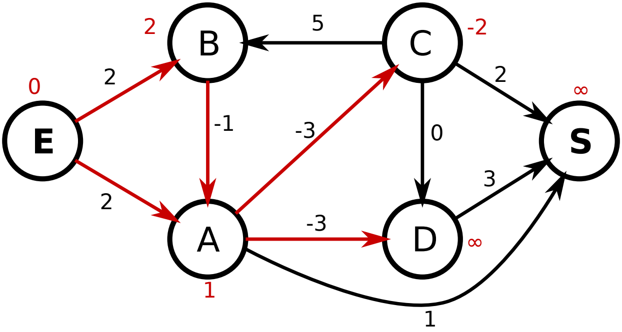
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



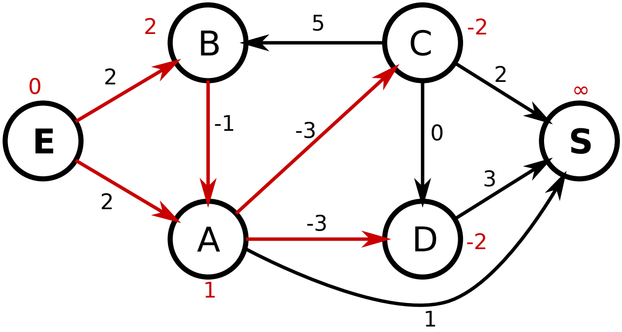
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



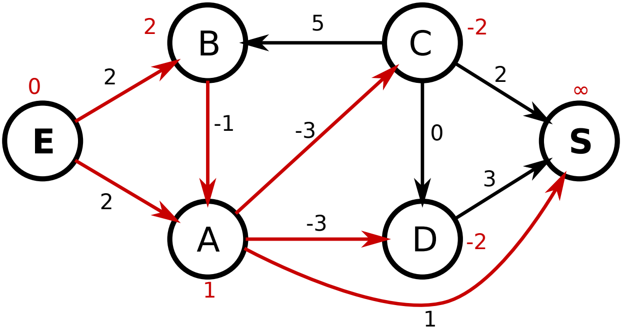
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



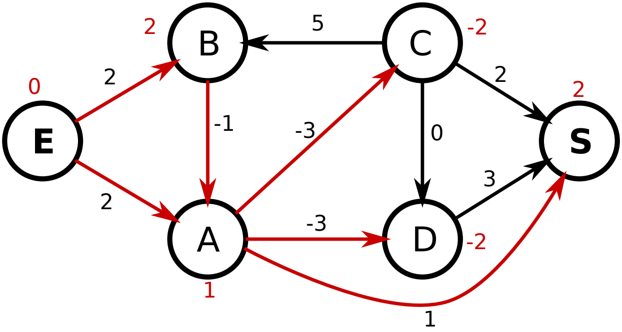
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



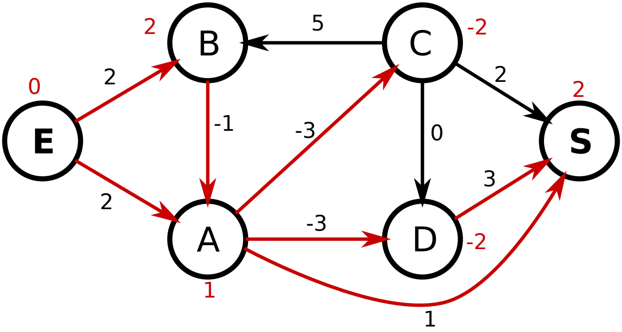
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



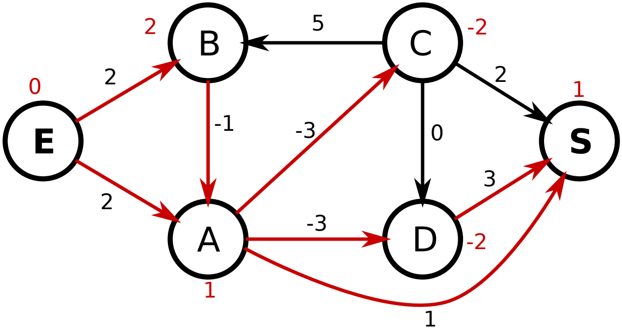
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



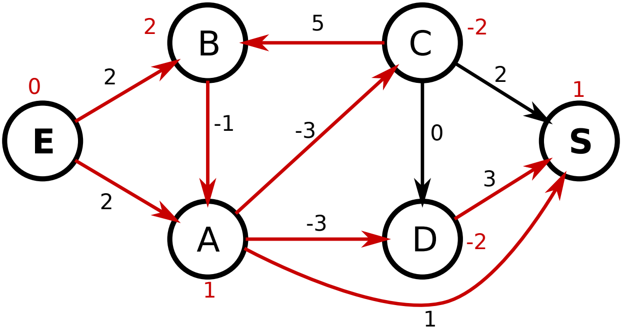
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



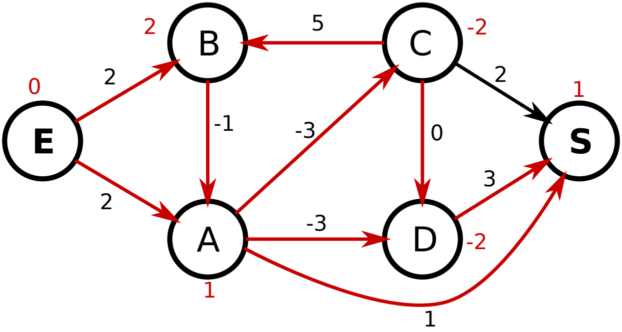
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



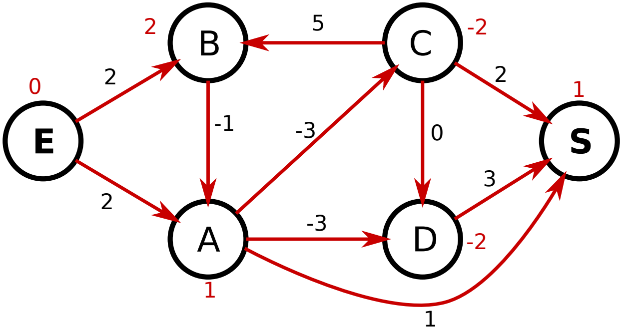
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



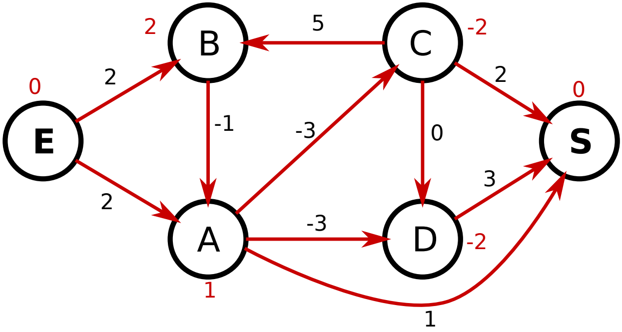
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



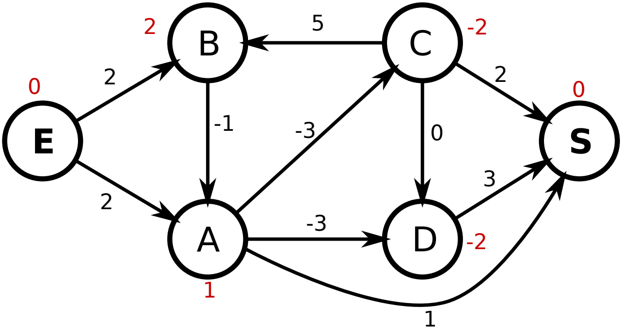
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



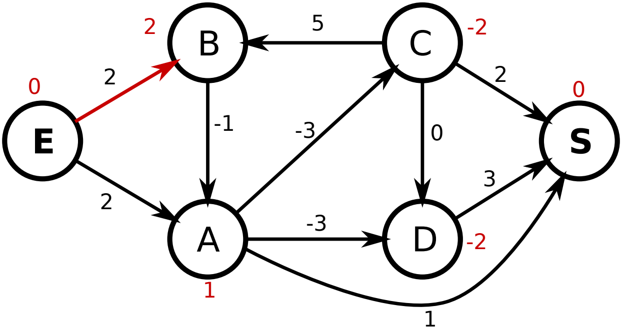
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



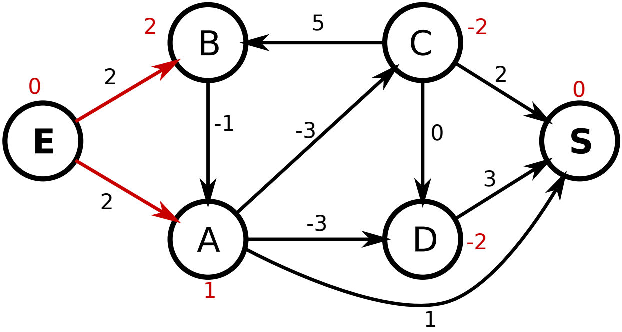
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



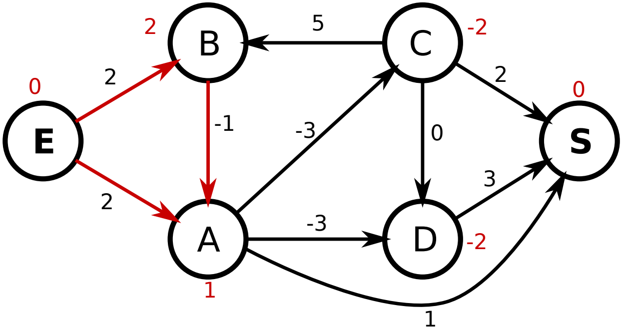
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



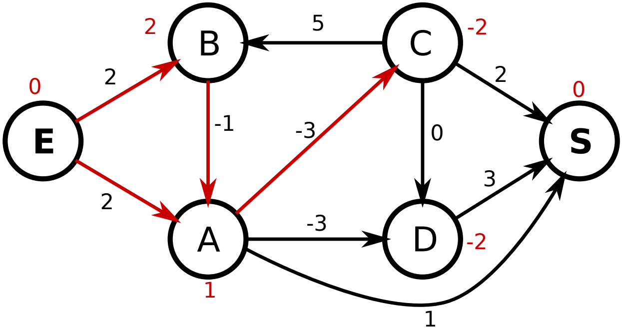
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



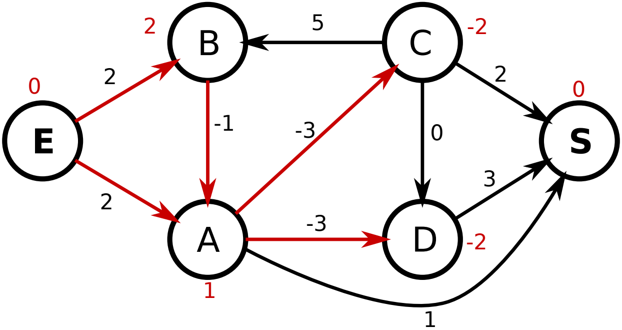
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



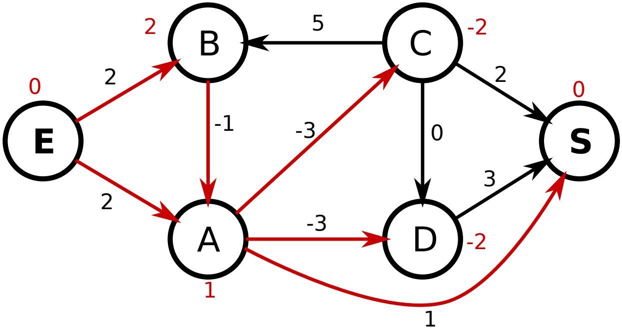
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



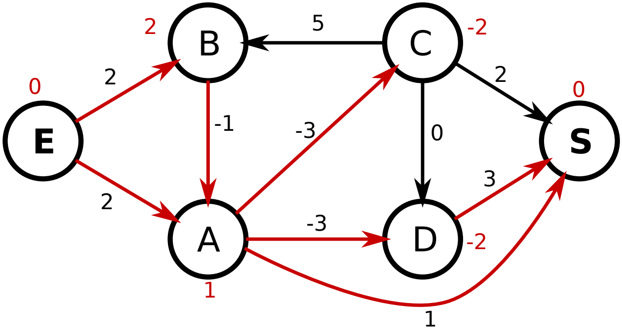
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



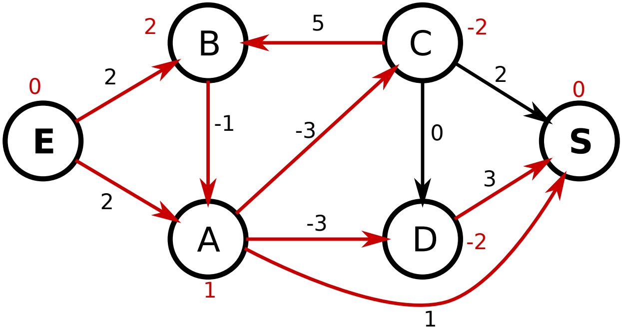
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



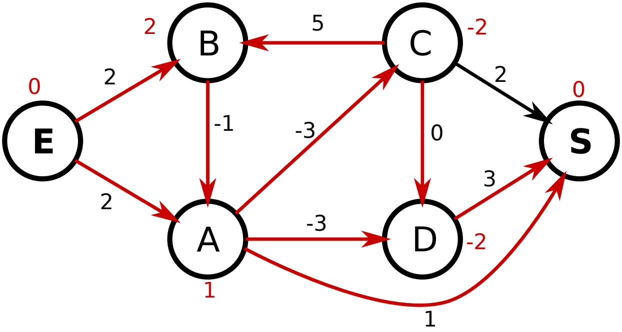
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



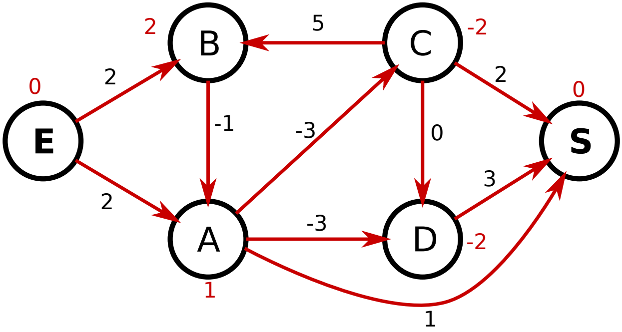
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



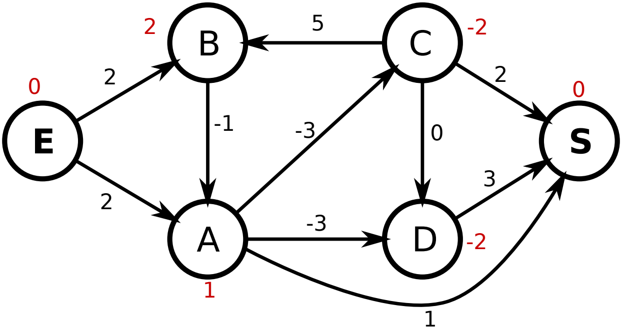
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



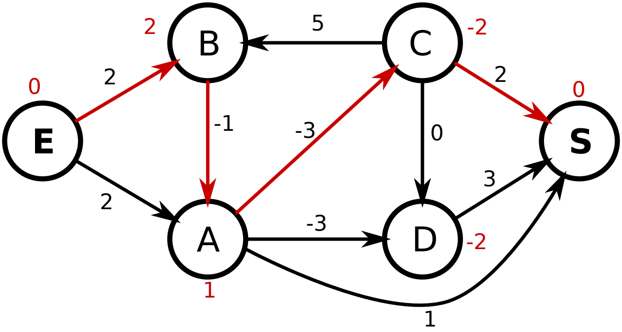
Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



Résolution

- Je construis le graphe
- J'applique l'algorithme de Bellman-Ford
- Le plus court chemin me permet de sortir sans rien dépenser



Exercice

L'administration d'une université doit organiser les examens d'une promotion d'étudiants. Il y a 7 options pour lesquelles un examen est nécessaire. Les étudiants de cette promotion sont répartis en 6 groupes selon les options suivies.

- Les étudiants du groupe A suivent les options 1, 2, 5
- Les étudiants du B suivent les options 2, 4, 5
- Les étudiants du C suivent les options 4, 5, 6
- Les étudiants du D suivent les options 2, 3
- Les étudiants du E suivent les options 3, 5
- Les étudiants du F suivent les options 1, 6, 7

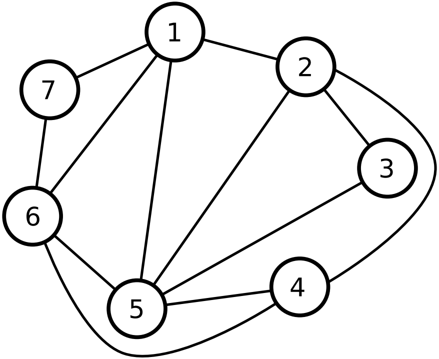
Les examens de chaque option ont la même durée. Il est évident que deux options qui sont suivies par les mêmes étudiants ne peuvent pas avoir leurs examens au même moment. L'objectif de l'administration est de minimiser le nombre de créneaux horaires nécessaires pour placer tous les examens. Proposez une solution en détaillant les étapes de votre réflexion et en justifiant chaque choix.

Comment représenter ce problème sous forme de graphe ?

- Un sommet représente l'examen d'une option
- Une arête représente le fait que deux options ne peuvent pas avoir lieu en même temps parce qu'il existe des étudiants qui suivent ces deux options.

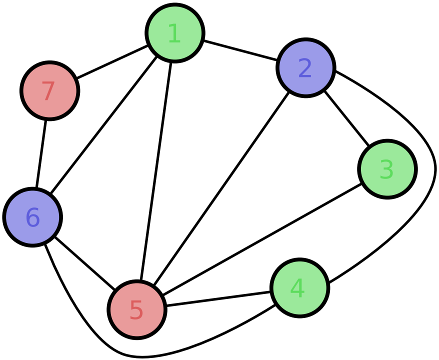
Quel est le problème ?

- Deux examens peuvent avoir lieu en même temps si aucun étudiants ne suit les deux options correspondantes
- Sur le graphe, cela signifie que les sommets représentant ces examens ne sont pas reliés
- Un ensemble de tels sommets est appelé un stable
- Il s'agit ici d'un problème de partition en stables, ou problème de coloration



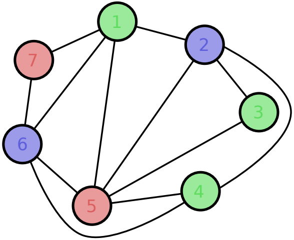
Résolution

- On construit le graphe
- On observe que le graphe est planaire, donc il existe une solution avec 4 créneaux horaires (c'est le théorème des quatre couleurs)
- Ce graphe contient un (au moins) cycle impaire, donc il faudra au moins 3 couleurs pour le colorier
- Il existe une solution avec 3 couleurs



Résolution

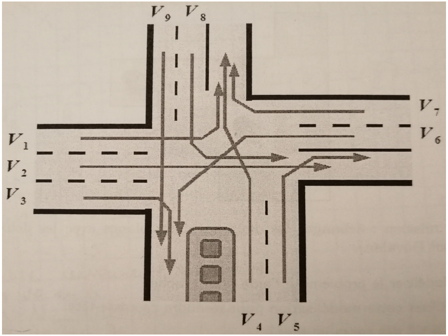
- On construit le graphe
- On observe que le graphe est planaire, donc il existe une solution avec 4 créneaux horaires (c'est le théorème des quatre couleurs)
- Ce graphe contient un (au moins) cycle impaire, donc il faudra au moins 3 couleurs pour le colorier
- Il existe une solution avec 3 couleurs



Créneaux horaires

- Les étudiants du groupe A suivent les options 1, 2, 5
- Les étudiants du B suivent les options 2, 4, 5
- Les étudiants du C suivent les options 4, 5, 6
- Les étudiants du D suivent les options 2, 3
- Les étudiants du E suivent les options 3, 5
- Les étudiants du F suivent les options 1, 6, 7

Une couleur correspond à un créneau horaire, aucun étudiant n'a deux examens en même temps.



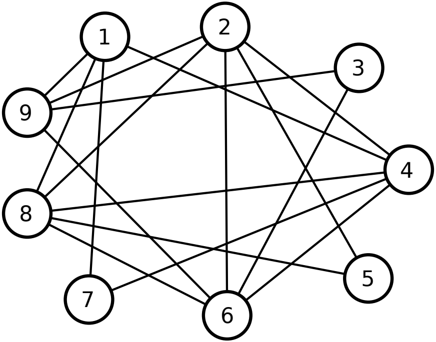
d'après Chartrand et Zheng

Contexte

- Image d'un carrefour avec ses voies de circulation
- Une voie ne doit pas avoir feu vert en même temps qu'une voie qu'elle croise ou qui va au même endroit
- Une phase de feu est un moment pendant lequel un certain nombre de voies ont feu vert en même temps
- Pendant que certaines voient sont au vert, d'autres attendent au rouge
- On souhaite modéliser ce carrefour afin de trouver le nombre de phases de feu minimum pour que le temps au feu rouge ne dure pas trop longtemps

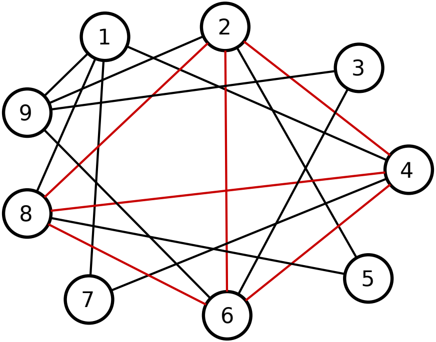
Éléments de réponse

- Un sommet est une voie de circulation
- Une arête représente le fait que deux voies ne peuvent pas avoir vert en même temps
- Un ensemble de voies qui peuvent avoir vert en même temps forment un stable dans le graphe
- Chercher les phases de feu revient à un problème de coloration dans ce graphe
- Les sommets 2, 4, 6, 8 forment une clique donc il faudra au moins 4 couleurs
- {7,8,9} ; {1,5,6} ; {2,3} ; {4} est une solution avec 4 couleurs



Éléments de réponse

- Un sommet est une voie de circulation
- Une arête représente le fait que deux voies ne peuvent pas avoir vert en même temps
- Un ensemble de voies qui peuvent avoir vert en même temps forment un stable dans le graphe
- Chercher les phases de feu revient à un problème de coloration dans ce graphe
- Les sommets 2, 4, 6, 8 forment une clique donc il faudra au moins 4 couleurs
- {7,8,9} ; {1,5,6} ; {2,3} ; {4} est une solution avec 4 couleurs



Éléments de réponse

- Un sommet est une voie de circulation
- Une arête représente le fait que deux voies ne peuvent pas avoir vert en même temps
- Un ensemble de voies qui peuvent avoir vert en même temps forment un stable dans le graphe
- Chercher les phases de feu revient à un problème de coloration dans ce graphe
- Les sommets 2, 4, 6, 8 forment une clique donc il faudra au moins 4 couleurs
- {7,8,9} ; {1,5,6} ; {2,3} ; {4} est une solution avec 4 couleurs