29 août 2018
Licence Parcours Info
Omar Abdelkafi
François Lemaire
Arnaud Liefooghe
Sylvain Salvati
Marie-Émilie Voge
Léopold Weinberg

Université Lille1 Sciences et Technologies

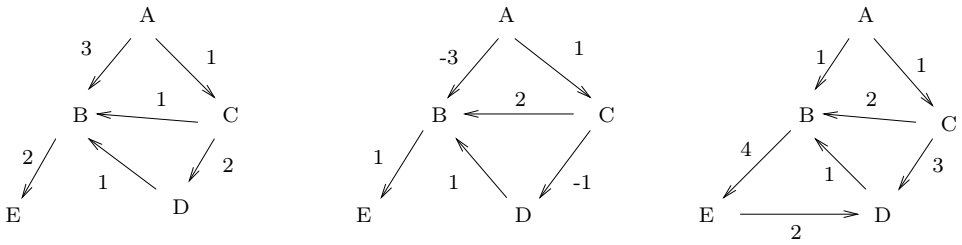
ARO

TD 9

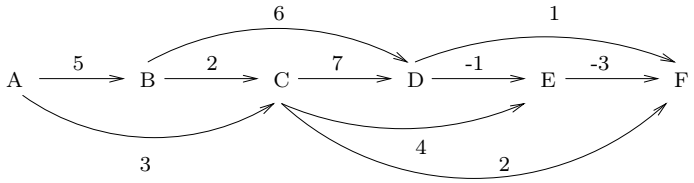
---

1 Algorithmes de Bellman et de Dijkstra

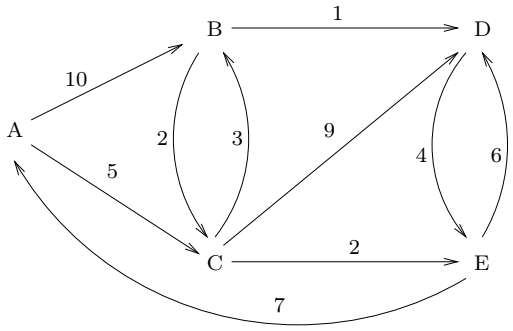
Q 1. On souhaite calculer un chemin de valeur minimale de A vers tous les autres sommets pour chacun des graphes suivants. Indiquer les algorithmes qui peuvent être utilisés dans chacun des cas.



Q 2. Appliquer l'algorithme de Bellman sur l'exemple ci–dessous, à partir du sommet A.



Q 3. Appliquer l'algorithme de Dijkstra sur le graphe suivant, à partir du sommet A. On suppose l'ensemble des sommets verts géré au moyen d'une file avec priorité implantée grâce à un tas binaire. Donner à chaque itération le contenu du tas.



Q 4. Appliquer l'algorithme de Dijkstra sur le graphe de la question 2. Que constate–t–on ?

2 Variations

Q 1. On souhaite adapter l'algorithme de Bellman pour qu'il calcule les chemins de valeur maximale entre un sommet s et tous les autres sommets d'un graphe G. Est–ce possible ? Si oui, comment faire ?

Q 2. Même question pour l'algorithme de Dijkstra.

Q 3. On considère un graphe valué G comportant éventuellement des circuits. Les valeurs des arcs représentent des débits. On cherche à calculer un chemin de débit maximum entre un sommet s donné et tous les autres sommets de G, sachant que le débit d'un chemin est le minimum des débits des arcs qui le composent.

Peut–on adapter l'algorithme de Dijkstra pour cela ? Si oui, le faire. Si non, pourquoi ?

Q 4. On considère un graphe valué G comportant éventuellement des circuits. Les valeurs des arcs représentent des probabilités. On cherche à calculer un chemin de plus grande probabilité entre un sommet s donné et tous les autres sommets de G, sachant que la probabilité d'un chemin est le produit des probabilités des arcs qui le composent.

Peut–on adapter l'algorithme de Dijkstra pour cela ? Si oui, le faire. Si non, pourquoi ?

Il est possible de réduire le problème en un problème de calcul de chemin de valeur minimale dans un graphe valué positivement en considérant les logarithmes des probabilités. Comment faire ?

3 Comptage de chemins dans un graphe acyclique

On se donne un graphe orienté sans circuit $G = (S, A)$ et un sommet distingué s à partir duquel tous les autres sommets de G sont accessibles. On veut mettre au point un algorithme qui, pour chaque sommet $x \in S$, donne le nombre $n(x)$ de chemins possibles de s vers x (on ne veut pas les chemins, seulement leur nombre).

Q 1. Que vaut $n(s)$?

Q 2. Exprimer $n(x)$ en fonction des $n(y)$ où y balaie l'ensemble des prédecesseurs de x.

Q 3. Donner en pseudocode, à la façon du cours, un algorithme fondé sur les idées précédentes, qui calcule $n(x)$ pour tous les sommets x du graphe. L'algorithme ne doit pas dépasser douze lignes.

Q 4. Déterminer la complexité en temps, dans le pire des cas, de l'algorithme de la question précédente. On peut supposer que le graphe est implanté avec des listes de prédecesseurs et des listes de successeurs. Si votre algorithme comporte plusieurs étapes successives, indiquer la complexité en temps, dans le pire des cas, de chacune des étapes.

4 Implantation de l'algorithme de Dijkstra

On s'intéresse à la version suivante de l'algorithme de Dijsktra. On suppose tous les sommets du graphe accessibles à partir de s.

```
procedure Dijkstra (G, s)
begin
  colorier tous les sommets sauf s en bleu
  colorier s en vert
  pi(s) := 0
  pred(s) est indéfini
  while il existe au moins un sommet vert do
    parmi les sommets verts, en choisir un, x, tel que pi(x) soit minimal
    colorier x en rouge
    for tous les successeurs y de x do
      if y est bleu ou (y est vert et pi(y) > pi(x) + v(x,y))) then
        colorier y en vert
        pi(y) := pi(x) + v(x,y)
        pred(y) := x
      fi
```

```
    od
  od
end
```

On souhaite implanter l'algorithme de Dijkstra au moyen d'une file avec priorité gérée au moyen d'un tas binaire. Le tas est constitué d'une structure à deux champs : un tableau T indicé de 0 à $n-1$ dont les éléments sont des sommets ; le nombre t de sommets présents dans la tas. La propriété de tas binaire maintenue est : *pour tout indice* $0 \leq i < t$, *le sommet en* T_i *a une valeur de* pi *inférieure à celle de ses deux fils, qui sont* T_{2i} *et* T_{2i+1}. Les primitives disponibles sont :

```
procédure vider_tas (tas)
fonction est_vide_tas (tas) retourne un booléen
fonction extraire_min_tas (tas) retourne T[1]
       note : le sommet est retiré du tas ; le tas est reconstruit.
procedure inserer_tas (sommet, tas)
       note : insère un nouveau sommet dans le tas
procedure pousser_tas (sommet, tas)
       note : la valeur de pi d'un sommet présent dans le tas est diminuée
              met-à-jour la position du sommet dans le tas.
```

Q 1. Préciser l'algorithme de Dijkstra en utilisant les primitives ci–dessus.

Pour que l'implantation de l'algorithme de Dijkstra ait une bonne complexité en temps dans le pire des cas, il faut, lorsque la valeur de *pi* d'un sommet x est diminuée, pouvoir déterminer très rapidement la position de x dans le tas (procédure *pousser_ tas*).

Q 2. Comment faire ?