Algorithmique — Maîtrise
TD 8
Arbres couvrants minimaux, jeu de Shannon, colorabilité

Isabelle Fagnot, Mihaela Sighireanu, Yan Jurski, Olivier Serre.

1 Deuxième meilleur arbre couvrant minimal

On considère un graphe non orienté $G = (V, E)$ muni d'une fonction de poids $w : E \to \mathbb{R}$. On suppose de plus que $|V| \leq |E|$ et que les poids de tous les arcs sont **distincts**.

Un *deuxième meilleur arbre couvrant minimal* est un arbre défini comme suit. Soit $\mathcal{T}$ l'ensemble des arbres couvrants de G et soit T' un arbre couvrant minimal de G. Un deuxième meilleur arbre couvrant minimal est un arbre couvrant T tel que

$$w(T) = \min\left(\{w(T'') \mid T'' \in \mathcal{T}\} \setminus \{w(T)\}\right)$$

1. Montrer qu'il existe un unique arbre couvrant minimal.
2. Montrer qu'il peut par contre y avoir plusieurs deuxièmes meilleurs arbres couvrants minimaux.
3. Soit T l'arbre couvrant minimum de G. Montrer qu'il existe deux arcs de E, $(u, v) \in T$ et $(x, y) \in T'$, tels que $T \setminus \{(u, v)\} \cup \{(x, y)\}$ est un deuxième meilleur arbre couvrant minimal de G.
4. Soit T un arbre couvrant de G. Pour tout $u, v \in V$, soit $\max[u, v]$ l'arc de poids maximal dans l'unique chemin de u à v dans T.
   Donner un algorithme fonctionnant en temps $\mathcal{O}(|V|^2)$ qui étant donné T calculer $\max[u, v]$ pour tout $u, v \in V$.
5. Donner un algorithme efficace de calcul d'un deuxième meilleur arbre couvrant minimal de G.

2 Jeu de Shannon

Soit $G = (X, E, u, v)$ un multigraphe non orienté (on autorise plusieurs arcs entre deux mêmes sommets), dont deux sommets u et v sont distingués. Un jeu de Shannon sur G se pratique à deux joueurs, l'un étant *attaquant* et l'autre *défenseur*. Chaque joueur colorie à tour de rôle une arête, l'attaquant en vert, le défenseur en rouge. L'attaquant gagne s'il parvient à construire un chemin entre u et v composé d'arêtes vertes, et perd sinon.

1. Pratiquer le jeu sur quelques exemples de graphes.

Les graphes peuvent être classés en trois catégories, les graphes *faibles* pour lesquels il existe une stratégie de l'attaquant lui permettant de gagner contre toute défense et quel que soit le joueur qui joue en premier, les graphes *résistants* pour lesquels il existe une stratégie du défenseur lui permettant de gagner même si l'attaquant joue en premier, et enfin les graphes *neutres*, pour lesquels il existe une stratégie gagnante pour le joueur qui commence, quel qu'il soit.

2. Donner quelques exemples de graphes de chacun de ces trois types.

3. En supposant qu'on dispose d'un algorithme permettant de tester si un graphe est faible, montrer qu'on peut en fait déterminer de quel type est un graphe donné.

Un résultat dû à A. Lehman décrit les graphes faibles : ce sont les graphes pour lesquels il existe un sous-ensemble Y de X, contenant u et v, et tel que le sous-graphe correspondant possède deux arbres couvrants disjoints (au sens des arêtes).

4. Proposer un algorithme gagnant pour l'attaquant lorsque le graphe vérifie cette condition.

3 Colorabilité

La coloration de graphes trouve une application naturelle dans le coloriage de cartes géographiques : le but est de colorier la carte en utilisant le moins de couleurs possible, en coloriant deux pays voisins avec des couleurs différentes. Le nombre minimal de couleurs nécessaires pour colorier un graphe G est appelé *nombre chromatique de* G, et noté $\chi(G)$.

1. Dessiner le graphe (non orienté) à huit sommets dont les arêtes sont :
   $(1,2),\ (1,3),\ (1,4),\ (1,7),$
   $(2,3),\ (2,4),\ (2,5),$
   $(3,5),\ (3,6),\ (3,7),$
   $(4,5),\ (4,7),$
   $(5,6),\ (5,7),\ (5,8),$
   $(6,8),$
   $(7,8).$

L'algorithme de Welsh-Powell donne une valeur approchée du nombre chromatique. Il consiste à :

- ordonner les sommets de G par ordre décroissant de leur degré.
- attribuer la couleur c_1 au premier sommet, puis dans l'ordre à chaque sommet qui n'est pas adjacent aux sommets précédemment coloriés en c_1.
- répéter l'opération avec la couleur c_2, et ainsi de suite jusqu'à ce que tous les sommets soient coloriés.

2. Justifier le fait que l'algorithme fournit une majoration de $\chi(G)$.
3. Appliquer l'algorithme au graphe donné précédemment.
4. En déduire que son nombre chromatique est 3.
5. Que peut-on dire de $\chi(K_n)$, où K_n est le graphe complet à n sommets ?
6. Préciser la complexité de l'algorithme de Welsh-Powell.

Le calcul du nombre chromatique d'un graphe est un problème NP-complet. Déterminer si un graphe est k-colorable n'est donc pas trivial. On sait cependant des choses sur les graphes planaires (donc sur les cartes géographiques), ainsi que sur les graphes bipartis[1] :

7. Montrer l'équivalence des trois propriétés suivantes :
   - G est 2-colorable.
   - G est un graphe biparti.
   - tous les cycles de G sont de longueur paire.

[1] C'est à dire un graphe tel qu'il existe une partition des sommets en deux sous-ensembles de sorte qu'il n'existe pas d'arc entre sommets du même ensemble

Correction 1 :

1. On suppose qu'il existe deux arbres couvrants minimaux T_1 et T_2. On regarde les arcs par ordre croissant de poids. On peut supposer que le premier arc absent dans un des arbres est dans T_1 et pas dans T_2. Soit (u, v) cet arc. Dans T_2, il existe un chemin de u à v. Si l'on ajoute (u, v) à ce chemin, on obtient alors un cycle. De plus ce cycle contient un arc de poids strictement (tous les poids des arcs sont distincts) supérieur au poids de (u, v) : en effet, il existe un arc dans le cycle qui n'est pas dans T_1 (car celui-ci contiendrait alors le cycle) et cet arc est donc de poids supérieur au poids de (u, v) (on prend les arcs dans l'ordre croissant). Si on enlève cet arc dans T_2 et que l'on ajoute l'arc (u, v), on obtient un nouvel arbre couvrant de G dont le poids est strictement inférieur à celui de T_2, ce qui est contradictoire avec la minimalité de T_2.
2. On pourra considérer l'exemple de la figure 1. L'arbre couvrant minimal est de poids 6. Les deuxièmes meilleurs arbres couvrants minimaux sont au nombre de 2 et sont de poids 10.

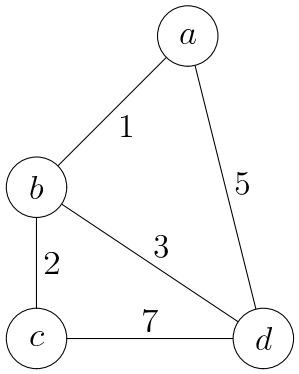

FIG. 1 – Un graphe avec deux deuxième meilleurs arbres couvrants minimaux

3. On raisonne un peu comme dans la première question. On appelle T l'arbre couvrant minimal et on appelle T' un deuxième meilleur arbre couvrant. On regarde les arêtes dans l'ordre croissant. On trouve alors une arête (u, v) qui appartient à l'un des deux arbres et pas à l'autre. Dans le second arbre, il y a un chemin de u à v et donc si l'on ajoute (u, v) on obtient un cycle. Comme ce cycle n'est pas dans le premier arbre et que l'on prend les arêtes dans l'ordre croissant, il existe une arête (x, y) du second arbre qui n'est pas dans le premier, et son poids est plus grand que celui de (u, v). En remplaçant (x, y) par (u, v) dans le second arbre, on obtient un arbre couvrant de poids strictement inférieur. Par minimalité de T, on en conclut donc que le second arbre est T'. Soit $T'' = T' \setminus \{(x, y)\} \cup \{(u, v)\}$. On a alors $w(T'') < w(T')$ et par minimalité de $w(T)$, $w(T) \leq w(T'')$. Or il n'existe pas d'arbre de poids strictement compris entre $w(T)$ et $w(T')$. On a donc $T'' = T$ et donc $T' = T \setminus \{(u, v)\} \cup \{(x, y)\}$
4. On donne un algorithme fonctionnant en temps linéaire pour calculer $\max[u, v]$ pour un u fixé et pour tout $v \in V$. Du coup l'algorithme demandé est l'appel de cet algo pour tout u et fonctionne dans le temps voulu.

   Pour réaliser le premier algo, on remarque que l'arc de poids maximal pour un

chemin $(v_0, v_1, \ldots, v_k)$ est l'arc de poids maximal pour le chemin $(v_0, v_1, \ldots, v_{k-1})$ ou l'arc v_k. On peut donc calculer tout cela à l'aide d'un parcours en profondeur. Cela donne alors :

```
Variable Globale
    Tableau d'arc M[|V|][|V|] initialisé à NIL

MAX()
    Pour tout sommet u dans V faire
        MAX-SOURCE(u,u)

MAX-SOURCE(s,u)
    Pour tout sommet v voisin de u dans T faire
        Si M[s,u]=NIL ou w(u,v) > w(M[s,u])
            M[s,v] = w(u,v)
        Sinon
            M[s,v] = M[s,u]
        MAX-SOURCE(s,u)
```

5. La troisième question montre qu'un deuxième meilleur arbre couvrant minimal peut être obtenu en ajoutant une arête à l'arbre couvrant minimal, ce qui crée un cycle, et à retirer une arête du cycle ainsi créé. Il reste bien sûr à déterminer la bonne arête à ajouter et la bonne arête à retirer. Etant donné l'ajout d'une arête (x, y) et le retrait d'une arête (u, v), l'augmentation du poids de l'arbre couvrant est de $w(x, y) - w(u, v)$. C'est cette quantité que l'on souhaite minimiser. Etant donné l'ajout d'une arête (x, y) les candidats possibles pour (u, v) sont toutes les arêtes de l'arbre couvrant minimal T reliant x à y : la plus intéressante est celle de poids maximal, c'est à dire $max[x, y]$. Il ne reste plus qu'à calculer pour toute arête (x, y) qui n'est pas dans T la valeur de $w(x, y) - max[x, y]$.
   On a alors la procédure suivante qui détermine un deuxième meilleur arbre couvrant minimal à partir de G, de l'arbre couvrant minimal T et de la matrice M précédemment calculée.

```
Deuxième_Meilleur()
    Variable dif initialisée à infini
    Variable e initialisée à NIL
    Pour toute arête (u,v) de E pas dans T faire
        si dif > w(u,v) - w(M[u,v]) alors
            dif = w(u,v) - w(M[u,v])
            e = (u,v)
    Si e = NIL retourner "le graphe est un arbre et il n'y a pas
    de second meilleur arbre couvrant"
    Sinon retourner T - e union M[e]
```

L'algo final commence donc par calculer un arbre couvrant minimal (Prim ou Kruskal) puis appelle MAX et enfin Deuxième_Meilleur.

Le temps total est donc celui du calcul de l'arbre couvrant minimal (les deux autres algo sont en $\mathcal{O}(|E|)$) et est donc en $\mathcal{O}(|E|\log|V|)$.

Correction 2 :

1.
2. 2 sommets connectés par une arc : c'est neutre.
   Deux sommets connectés par 2 arc c'est faible. La clique à 4 entre 2 voisins quelconques est faible.
   Le losange, est résistant si on joue pour la diagonale.
3. On commence par demander si le graphe est faible. Si c'est le cas, c'est terminé. Sinon, on a soit un graphe neutre soit un graphe résistant. Pour savoir s'il est faible, il suffit de savoir si l'attaquant à une stratégie quant il joue en premier (si c'est le cas, le défenseur en a une lorsqu'il commence, car sinon le graphe serait faible). Il reste alors à tester tous les premiers coups possibles pour l'attaquant et regarder si le graphe obtenu en fusionnant les deux sommets reliés par l'arc associé au premier coup est faible. Si on ne trouve pas un tel premier coup, on conclut que le graphe est résistant.
4. On commence par préciser la notion de résultat d'un coup dans le jeu. Lorsque c'est le défenseur qui joue le résultat de son coup est le graphe précédent privé de l'arc colorié en rouge par le défenseur. Lorsque l'attaquant joue, le résultat est le multigraphe obtenu en fusionnant les deux sommets liés par l'arc colorié en vert. A la fin, la partie est remporté par l'attaquant si u et v sont dans la même composante.
   On peut supposer que $Y = X$ (sinon on ignore tout ce qui a trait à $X \setminus Y$).
   On décrit maintenant une stratégie gagnante pour l'attaquant. On appelle T_1 et T_2 deux arbres couvrants disjoints.
   Si A ou D joue une arete qui n'est pas dans T_1, ni T_2, on la retire (on s'en fout en fait).
   Si c'est a A de jouer en premier, il fait ce qu'il veux, puis on fusionne le graphe. Cela peut modifier aussi T_1 et T_2, mais chacun reste couvrant, donc peu importe.
   Si c'est D qui joue, par exemple sur T_1, alors le "resultat" deconnecte T_1 en T_1' et T_1''. A va essayer de corriger ce Pb en les reconnectant. Soit x dans T_1' et y dans T_1'', puisque T_2 est couvrant, il existe un chemin dans T_2 entre x et y. Le premier arc de ce chemin qui sort de T_1' va dans T_1'', A joue cet arc, il les reconnecte.
   A tout moment après un coup de A, on a deux arbres couvrants disjoints de la composante connexe contenant u et v. A la fin, il n'y a plus d'arc et donc u et v sont fusionnés. Il reste à dire que D ne peut déconnecter u et v car sinon il n'y aurait pas deux arbres couvrants disjoints.

Correction 3 :

1.
2. Il suffit de prouver que l'algorithme propose bien un coloriage du graphe, ce qui est immédiat.
3.
4. Il faut justifier que moins de 3 on ne peut pas. Par exemple en exhibant un motif : boucle de longueur 3.

5. Vous devriez réussir à colorier K_n avec n couleurs. Si l'on suppose qu'il existe une coloration avec moins de n couleurs, alors il y a 2 sommets qui ont la même couleur, or le graphe est complet, donc il y a un arc entre ces deux sommets ....
6. $O(|V|^2 + |E|)$ pour la dernière partie de l'algo, c'est a dire $O(|V|^2)$: pour chaque sommet que l'on colorie on peut ne parcourir que ses voisins.
7. 2-colorable $\Leftrightarrow$ biparti est facile, les couleurs fournissant la partition et vice-versa.

   2-colorable $\rightarrow$ cycles pairs c'est facile. Voici une redaction : soit $s_0...s_n$ un cycle. Disons que le graphe est colorie avec des 0 et des 1. Alors $c(s_{i+1}) = c(s_i) + 1 mod 2$. Si on fait la somme de toutes ces égalités, on a $c(s_0) = c(s_n) + n mod 2$ et comme $s_0 = s_n$ (cycle), on a $n = 0 mod 2$.

   Cycles pairs $\Rightarrow$ 2 Coloriable, c'est plus dur. Il faut chercher à colorier le graphe. Fixons un s_0 de couleur 0 et on propage la couleur aux voisins. Bref, on se pose la question de ce qui se passe quand il y a plusieurs façon d'atteindre un sommet en partant de s_0. Montrons que tout les chemins de s_0 a x quelconque ont la même parité : soit γ_1, γ_2 deux tels chemins, $\gamma_1.\tilde{\gamma_2}$ est un cycle, donc pair, donc γ_1 et γ_2 ont la même parité. On a montré : si on part de s_0, et qu'on fait un parcours (profondeur ou largeur peut importe), on peut donner une coloration de proche en proche qui est unique. On a pas dit qu'elle était correcte, mais elle l'est : si x et y voisins ont la même couleur c'est qu'il existe un chemin γ de s_0 à x et μ de s_0 à y de même parité (à cause de la même couleur), or $\gamma.(x,y).\tilde{(\mu)}$ est un cycle, et il faudrait qu'il soit de longueur paire avec $\gamma.\tilde{(\mu)}$ pair, impossible. OK ma coloration est une vrai coloration.

   Au passage on remarquera que tout cela donne un algo pour 2-colorier un graphe si c'est possible.