TD 4: Min-cut randomisé algorithme de Krager

Université Joseph Fourier
UFR IM²AG

Concept : Génération aléatoire, contrôle de la probabilité d'échec
Méthode : algorithme glouton, randomisation

Auteur: Nicolas Gast

L'objectif de cette séance est d'étudier un algorithme randomisé résolvant le problème Coupe-Min (MinCut). Ce problème peut être résolu par un algorithme polynomial déterministe en cherchant un flot maximum (p.e., en utilisant l'algorithme de Ford-Fulkerson). Dans ce TD, nous allons explorer une technique se basant sur un générateur aléatoire, introduite par Karger en 1993. On obtient ainsi un algorithme dit randomisé, *qui fournit un bon résultat avec une probabilité que l'on peut estimer.*

Soit $G = (S, A)$ un graphe non orienté, connexe , ayant $|S| = n$ sommets. Une coupe $C \subset A$ est un ensemble d'arêtes de G tel que, si on enlève ces arêtes, le graphe partiel $G'' = (S, A \setminus C)$ a 2 composantes connexes. Le problème de la coupe minimale est de trouver une coupe ayant un minimum d'arêtes.

On étend la notion de graphe à celle de multigraphe comme étant un ensemble de sommets S et une liste d'arêtes (couples de sommets). C'est à dire qu'entre deux sommets, on peut avoir plusieurs arêtes. On définit la contraction d'un multigraphe $G = (S, M)$ par une arête (x, y) comme étant le multigraphe $G' = (S', M')$ dans lequel :

1. on remplace les sommets x et y par un sommet z ;
2. on supprime les arêtes entre x et y ; et
3. on remplace les arêtes de la forme (x, u) ou (y, u) par (z, u).

Voir par exemple la Figure 1.

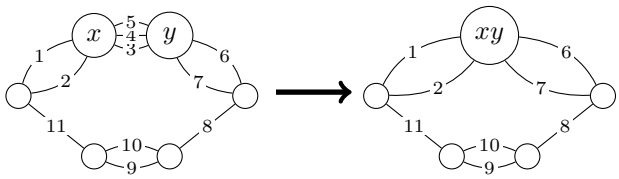

FIGURE 1 – Contraction du multigraphe par l'arête 3 reliant x et y.

Exercice 1: Génération d'une coupe aléatoire

On se donne l'algorithme suivant :

Génère_coupe(G)
Input : multigraphe G
Output : G contient 2 sommets, les arêtes entre ces sommets forment une coupe du multigraphe G initial
for $i = 1$ **à** $n - 2$ **do**
  Choisir une arête a uniformément dans le multigraphe G ;
  $G \leftarrow$ contraction(G, a) ;
Retourner G ;

Algorithme 1 : Génération d'une coupe aléatoire.

1. Expliquer le fonctionnement de l'algorithme 1 sur l'exemple de la Figure 1.
2. Comme travail personnel (à la maison), montrer que cet algorithme génère bien une coupe.
3. À partir d'exemples exprimer votre intuition que la coupe générée ne doit pas être trop mauvaise.

Exercice 2: Propriétés de la coupe aléatoire

Pour analyser la complexité de cet algorithme estime la probabilité qu'il fournisse une coupe minimale. Le calcul exact étant difficile, nous allons minorer cette probabilité et montrer qu'elle est suffisante pour avoir un algorithme efficace.

Notations : on suppose que la taille minimale d'une coupe est k ; on choisit une coupe minimale C_{min} (on pourrait en avoir plusieurs). On note p la probabilité que l'algorithme 1 génère la coupe C_{min}.

1. Donner un minorant du nombre d'arêtes du graphe G.
2. Calculer un minorant de la probabilité de ne pas tomber sur une arête de C_{min} au premier tirage aléatoire.
3. Donner un minorant du nombre d'arêtes du multi-graphe graphe G' après la première contraction. Minorer la probabilité de ne pas tomber sur une des arêtes de C_{min} au second tirage.
4. Procéder par récurrence pour minorer la probabilité de ne pas tomber sur une des arêtes de C_{min} au $i^{\text{ème}}$ tirage.
5. Calculer un minorant de la probabilité que l'algorithme 1 génère la coupe minimale C_{min}.

Exercice 3: Design d'un algorithme qui garantit une probabilité d'erreur

Comme on ne génère pas "la bonne coupe" à tous les coups, on répète le tirage et on garde le meilleur résultat. Cela revient à générer un échantillon de taille suffisante T.

```
Répétition_coupe(G,T)
  Input : multigraphe G = (S, M), entier T (Taille de l'échantillon)
  Output : Meilleure coupe obtenue
  C_min ;
  for i = 1 à T do
      C_courant ← Génère_Coupe(G) ;
      if |C_courant| < |C_min| then
          C_min ← C_courant
  Retourner C_min;
```

Algorithme 2 : Répétition de tirages

1. Minorer la probabilité que l'algorithme 2 fournisse le bon résultat en fonction de T.
2. Montrer que pour une taille d'échantillon de l'ordre de $T = n^2$, la probabilité d'obtenir une coupe minimale est minorée par $1 - 1/e$.
3. On se donne une probabilité α et un graphe de taille n ; donner la taille d'échantillon telle que la probabilité d'avoir généré une coupe minimale soit supérieure à α. Donner la valeur de la taille de l'échantillon pour $\alpha = 0.99$ et $n = 100$.