Algorithmique et Programmation
Projet : Algorithme d'Edmonds pour les couplage maximaux dans un graphe

Ecole normale supérieure
Département d'informatique
td-algo@di.ens.fr

2013-2014

Dans un graphe arbitraire, un couplage est un ensemble d'arêtes dont les extremités sont toutes distinctes. Un couplage maximal est un couplage contenant le plus grand nombre d'arêtes possible. Dans les graphes bipartis, calculer un couplage maximal est relativement facile, en utilisant des algorithmes de flot maximal. C'est d'ailleurs assez bien connu, et discuté en détail dans de nombreux ouvrages, dont l'incontournable [**?**].

Dans les graphes généraux, les couplages maximaux sont bien définis, et il est toujours possible de les calculer en temps polynomial par un algorithme dû à Edmonds en 1965, mais qui est bien moins connu.

Comme dans l'algorithme de couplage maximal sur les graphes bipartis , il s'agit de construire des couplages de plus en plus grand en ajoignant au précédent un chemin augmentant.

1 Terminologie

Étant donné un couplage M, un sommet est dit libre s'il ne touche aucune arête de M. Un chemin dans le graphe est dit alternant si ses arêtes sont alternativement dans M et hors de M (donc si une arête sur deux est dans M). Un chemin augmentant est un chemin alternant dont les points de départ et d'arrivée sont deux sommets libres. On peut démontrer (et on admettra) qu'un couplage est maximal s'il n'existe pas de chemin augmentant par rapport à lui (c'est aussi vrai dans les graphes biparti que dans les graphes ordinaires). On a donc un algorithme simple pour le couplage maximal :

function MAXIMUM-MATCHING(G)
  $M \leftarrow \emptyset$
  loop
    $P \leftarrow$ AUGMENTING-PATH(G, M)
    if $P = \emptyset$ **then return** M **else** $M \leftarrow M \oplus P$
  end loop
end function

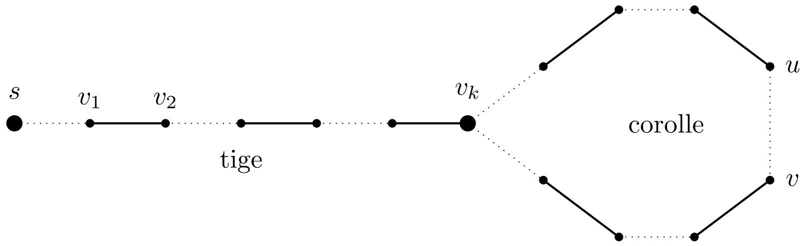
Ici, $M \oplus P$ désigne la différence symétrique des deux graphes (c'est-à-dire les arêtes qui sont dans l'un ou l'autre mais pas dans les deux). Le problème reste donc de trouver des chemins augmentants. Autant dans un graphe bipartite c'est assez simple à faire avec un simple parcours en profondeur, autant dans les graphes généraux c'est un peu plus compliqué.

Dans un graphe, l'opération de contraction d'une arête $u \leftrightarrow v$ consiste à retirer l'arête en question et à *fusionner* les deux sommets u et v. Le sommet résultant est adjacent aux voisins de u et de v.

Une fleur est constituée par l'ensemble des organes de la reproduction et des enveloppes qui les entourent chez certains végétaux. Pour ce qui nous intéresse, une fleur est composée d'une corolle, qui est la partie de la fleur formée par l'ensemble de ses pétales, et d'une tige. Je vous invite à consulter Wikipédia pour plus de détails botaniques

2 Idée sous-jascente : la contraction des fleurs

Étant donné un graphe G, un couplage (pas forcément maximal) M de G, et un sommet libre s, une fleur est formée d'un chemin alternant de longueur paire (éventuellement nulle) partant d'un sommet libre s et allant jusqu'à un sommet v_k (c'est la tige), et d'un chemin alternant de longueur impaire partant de v_k et revenant à v_k (c'est la corolle). Le sommet v_k est la base de la corolle.



Dans le cas où la tige n'est pas triviale, alors l'arête de la tige qui touche la base de la corolle est nécessairement la *seule* arête du couplage qui relie un sommet de la corolle à un sommet extérieur. On considère le graphe G' obtenu en contractant toutes les arêtes d'une corolle (qu'on va appeler C), en appelant c' le sommet résultant de la contraction de C, et M' le couplage de G' obtenu de la même manière. On démontre (parce que c'est utile pour écrire l'algorithme) que

Theorème 1. *S'il existe dans G' un chemin augmentant pour M', alors il existe dans G un chemin augmentant pour M'.*

Démonstration. On le construit explicitement, en "décompressant" la corolle et en adaptant le chemin de G'.

- Si le chemin ne passe pas par c', il n'y a rien à faire.
- Si c' est une des extrémités du chemin (disons $\cdots \leftrightarrow x \leftrightarrow c'$), alors l'arête $x \leftrightarrow c'$ n'appartient pas à M'. De plus, c' est nécessairement libre par rapport à M' (sinon le chemin de G' ne serait pas augmentant). Ce dernier fait implique qu'aucune arête du couplage M ne touche la corolle C, et donc en particulier que la base de la corolle est libre par rapport à M (donc que la tige est triviale). Pour étendre le chemin à l'intérieur de la corolle, on choisit un des sommets de C adjacents à x (disons y), et on part dans la corolle depuis y, en commençant par suivre l'arête du couplage M qui touche y. On s'arrête quand on atteint la base de la corolle car on a atteint un sommet libre. On a donc étendu le chemin augmentant de M' en un chemin augmentant de M.
- Si le chemin traverse c' (disons qu'il fait $\cdots \leftrightarrow x \leftrightarrow c' \leftrightarrow z \leftrightarrow \dots$), alors il y a une des deux arêtes en contact avec c' qui fait partie du couplage M' (la situation est complètement symétrique, donc on peut supposer sans perte de généralité que c'est $x \leftrightarrow c'$). Une fois qu'on aura décompressé C, on aura donc une arête du couplage entre x et la corolle, ce qui signifie (cf. supra) qu'elle doit toucher la base de la corolle. En fait, dans ce cas x est forcément le sommet d'avant v_k sur la tige. Partant de là, on fait comme dans le cas précédent pour brancher y sur la base de la corolle.

□

En gros, on va explorer le graphe à la recherche d'un chemin augmentant. Si on en trouve un, c'est bon. Si on trouve une fleur, on la contracte et on recommence récursivement l'exploration sur un graphe plus petit. Si on ne trouve ni fleur, ni chemin augmentant, c'est que le couplage actuel est maximal. Vu de suffisamment haut, ce qu'on va réaliser est :

function AUGMENTING-PATH(G, M)
    Explorer le graphe
    if chemin augmentant trouvé **then stop**
    if corolle trouvée **then**
        Générer G' en contractant la corolle
        $P \leftarrow$ AUGMENTING-PATH(G', M')
        if $P = \emptyset$ **then return** $\emptyset$ **else return** DÉCOMPRESSION(P)
end function

3 Exploration du Graphe

Pour explorer le graphe, on utilise une procédure de marquage. On commence avec un graphe sans étiquettes.

Signification des marques. Les sommets reçoivent des étiquettes de la forme $[x, P, y]$ (resp. $[x, I, y]$) qui signifient "accessible depuis le sommet libre x par un chemin alternant de longueur paire (resp. impaire), dont le sommet précédent est y". Si la signification des étiquettes est respectée, alors on a les invariant suivants :

i) Un sommet marqué $[s, P, \emptyset]$ est forcément libre (et c'est précisément s).

ii) si $u[s, P, x] \leftrightarrow v[s, I, u]$, alors par définition l'arête $u \leftrightarrow v$ ne fait pas partie du couplage.

iii) si $u[s, I, x] \leftrightarrow v[s, P, u]$, alors par définition l'arête $u \leftrightarrow v$ fait partie du couplage.

iv) Plusieurs sommets “impairs” peuvent avoir le même “ancêtre” (c.a.d. le même sommet précédent dans le chemin alternant), mais deux sommets “pairs” ne peuvent avoir le même ancêtre, car les sommets impairs n'ont qu'un seul descendant possible (celui qui leur est couplé).

Méthode de marquage. L'algorithme procède en marquant les sommets et les arêtes jusqu'à ce qu'on puisse aboutir à une conclusion. A chaque étape, on peut au choix :

1. Choisir un sommet libre et non-étiqueté s, et le marquer avec l'étiquette $[s, P, \emptyset]$
2. Ou bien choisir une arête non-examinée $u \leftrightarrow v$, où u porte une étiquette “paire”, disons $[s, P, x]$. On marque l'arête comme “examinée”, et on distingue plusieurs cas :

   A. Si v est libre et non-étiqueté : un chemin augmentant (entre s et v) a été trouvé. Il suffit de suivre les “parents” de v jusqu'à s pour obtenir la liste des noeuds du chemin.

   B. Si v est non-étiqueté mais couplé à w : donner l'étiquette $[s, I, u]$ à v, et l'étiquette $[s, P, v]$ à w. Continuer le marquage.

   C. Si v est étiqueté $[r, P, y]$ avec $r \neq s$: un chemin augmentant (entre r et s) a été trouvé. Il faut remonter les parents jusqu'à de v jusqu'à s d'un côté et jusqu'à r de l'autre pour établir la liste des sommets du chemin.

   D. Si v est étiqueté $[s, P, x]$: on a trouvé une corolle (cf. la figure au-dessus). Justification : u et v sont tous les deux accessibles depuis s par un chemin alternant de longueur paire. Si on écrit ces deux chemins, ils ont forcément un préfixe commun (ne serait-ce que s). Notons $s, v_1, \dots, v_k$ le plus grand préfixe commun, tout en notant que $k = 0$ est possible. Alors, v_k est forcément étiqueté “pair”, car il a deux “descendants”. Il s'ensuit qu'il existe un chemin alternant de longueur paire entre v_k et u d'une part, et entre v_k et v d'autre part. Dans ces deux chemins, l'arête qui part de v_k n'appartient pas au couplage, et l'arête qui arrive sur u (resp. v) appartient au couplage. Seulement, on a découvert une arête (hors couplage, forcément) entre u et v. Cet arête réalise un cycle alternant de longueur impaire qui part de v_k, passe par u, par v, et revient sur v_k. De plus, v_k est la base de la corolle. La tige, si elle existe, est le chemin alternant de longueur paire entre s et v_k.

En fait, le processus de marquage construit une forêt qui regoupe des sous-arbres de G “par dessus” G (un peu comme dans l'algorithme de Kruskal pour trouver un arbre couvrant minimal).

4 Travail demandé

Vu la complexité de l'algorithme, il s'agit avant tout de produire une implémentation *correcte* (ne plante pas, calcule le bon résultat), même si elle n'est pas extrêmement efficace. Une complexité de $\mathcal{O}\left(V^4\right)$ est admissible (il est possible de faire $\mathcal{O}\left(V^3\right)$ mais ça complique encore un peu les choses).

À chaque étape, testez vos implémentation avec d'assez grands exemples générés aléatoirement.

1. Implémenter un algorithme qui retourne la différence symmétrique d'un couplage et d'un chemin augmentant (tous deux donné en entrée).
2. Implémenter un algorithme qui trouve un chemin augmentant dans un graphe biparti.

   Tester cet algorithme sur des graphes bipartis pour voir qu'il trouve bien un couplage maximum (en le combinant avec l'algorithme de la question précédente).
3. Implémenter un algorithme qui prends en entrée un graphe G et un sous ensemble (connexe) de sommets de G et construct un nouveau graphe $H = G/S$ où S est contracté à un seul sommet s.
4. Implémenter un algorithme « de décompression » qui prends en entrée un graphe G, un couplage M dans G, une corolle S dans G et un chemin augmentant C dans le graphe contracté G/S (le chemin est augmentant par rapport aux arêtes restantes de M dans G/S) et qui retourned un chemin augmentant dans G.
5. Implémenter un algorithme qui prends en entrée un graphe G et un couplage M dans G et qui retourne soit un chemin augmentant, soit la corolle d'une fleur ou soit « M est maximum ».
6. Implémenter l'algorithme d'Edmonds pour trouver un couplage maximum dans un graphe non-biparti.