

INF431

Premiers exercices d'algorithmique
Sujet proposé par Bruno Salvy & Gilles Schaeffer

Version: 1475:1512M

1 Variations sur les mariages stables

À côté de l'interprétation des couplages C comme ensemble de paires d'éléments, on utilisera dans ces questions aussi la notation $C(x)$ pour désigner le ou la fiancée de x dans le couplage C (avec $C(x) = x$ si x n'est pas fiancé).

1.1 Stabilité

1.1.1 Préférence de caste

On veut affecter n candidats dans m services d'une administration. L'algorithme de Gale et Shepley permet de trouver une affectation lorsque $m = n$, chaque service classe les n candidats selon ses propres critères et chaque candidat classe les n services selon ses préférences.

Question 1 Supposons maintenant que le i ème service ait toujours une unique liste de préférence mais qu'on doive lui affecter n_i candidats (avec $\sum_{i=1}^m n_i = n$). Comment modifier l'algorithme pour résoudre ce nouveau problème d'affectation ? \diamond

Supposons que notre administration soit noyauté par les anciens élèves d'une célèbre école (les Y) : chaque service désire avant tout recruter un Y pour bénéficier de son réseau d'amitiés, et classe donc systématiquement les Y devant les autres candidats. Inversement tous les candidats classent les postes de direction devant les postes d'exécutants.

Question 2 Montrer que s'il y a k postes de direction et k candidats qui sont Y, alors chaque poste de direction est occupé par un Y. \diamond

1.1.2 Préférences incomplètes

L'expérience montre qu'il existe des individus qui préfèrent rester célibataire que d'être marié à une personne qui ne leur plaît pas suffisamment... On modélise cela en demandant à chaque homme ou femme de se classer dans sa liste de préférence (éventuellement en dernière position). Ceci amène à étendre la définition de stabilité aux couplages non nécessairement parfaits. Un couplage est instable s'il contient des éléments $h \in H$ et $f \in F$ qui se préfèrent mutuellement à leurs fiancés respectifs dans le couplage, ou s'il contient un élément $x \in H \cup F$ qui préfère rester célibataire qu'épouser son ou sa fiancé(e) dans le couplage. De manière équivalente l'instabilité du couplage revient à l'existence d'une paire $(x, y) \in (H \times F) \cup \{(x, x) \mid x \in H \cup F\}$ telle que $y >_x C(x)$ et $x >_y C(y)$.

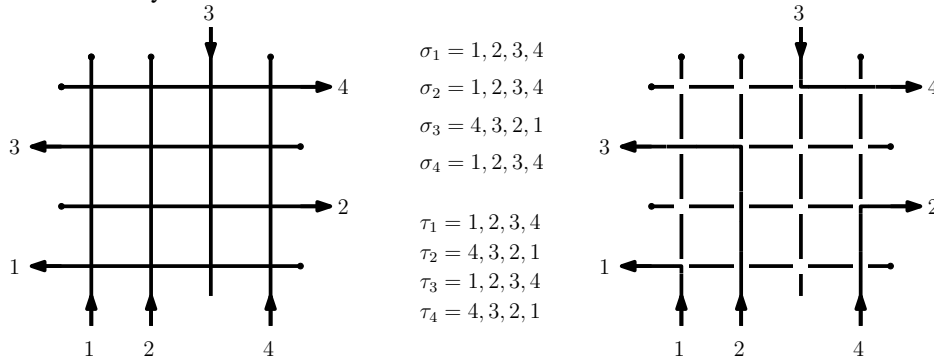
Question 3 Montrer que dans un couplage stable personne ne peut être marié à un individu qui apparaît plus bas que lui-même dans sa liste de préférence. \diamond

Cette propriété implique qu'on peut tronquer la liste de chaque individu à partir de la position où il apparaît : on parle de liste incomplète, et on dit que x apprécie y si y apparaît dans sa liste tronquée.

Question 4 Étendre l'algorithme de Gale et Shepley aux listes incomplètes, et montrer que les preuves de terminaison et de stabilité du couplage obtenu s'adaptent immédiatement. \diamond

1.1.3 Stabilité et routage parfait

On considère un réseau formé de n tuyaux d'arrivée et n tuyaux de sortie : le tuyau d'arrivée numéro i croise successivement à partir de l'entrée chacun des n tuyaux de sortie suivant une permutation $\sigma^{(i)}$; inversement, le tuyau de sortie numéro j est croisé par chacun des n tuyaux d'entrée et on note par une permutation $\tau^{(j)}$ l'ordre dans lequel ces tuyaux sont rencontrés en partant de la sortie. Chaque croisement est constitué d'un robinet qui peut être soit fermé, soit ouvert dans l'une des 6 positions connectant 2 des 4 tuyaux incidents.



Question 5 Montrer qu'on peut ouvrir les robinets de façon à permettre l'écoulement simultané de n flots indépendants dans le réseau, issus des n tuyaux d'entrée et s'écoulant par les n tuyaux de sortie. \diamond

1.2 Terminaison

On considère un groupe de n personnes, dont certaines sont amies et d'autres ennemies. Le but est de diviser le groupe en 2 équipes de sorte que chacun se sente bien dans son équipe, au sens où un individu x est content si

$$\text{Contentement}(x) \geq \text{Mécontentement}(x) \quad (1)$$

avec

$$\begin{aligned} \text{Contentement}(x) &= \#\{\text{amis de } x \text{ dans son équipe}\} + \#\{\text{ennemis de } x \text{ dans l'autre équipe}\} \\ \text{Mécontentement}(x) &= \#\{\text{ennemis de } x \text{ dans son équipe}\} + \#\{\text{amis de } x \text{ dans l'autre équipe}\} \end{aligned}$$

L'algorithme que nous voulons étudier fonctionne itérativement à partir d'une partition arbitraire : tant qu'il existe des personnes mécontentes, on en choisi une et on la change d'équipe.

Question 6 Donner un exemple de situation où une étape de l'algorithme fait augmenter le nombre de personnes mécontentes. \diamond

Question 7 Montrer que l'algorithme termine en produisant un groupe stable. \diamond

Question 8 Peut on vraiment utiliser cet algorithme pour constituer des équipes ? \diamond

(Cet algorithme a en fait été proposé pour modéliser l'autostabilisation d'un modèle simplifié de réseau de neurone appelé réseau de Hopfield : chaque individu représente un neurone pouvant prendre deux états, les connexions sont de deux types, favorisant l'accord ou le désaccord entre neurones voisins).

2 Le problème k -somme

De nombreux problèmes de décision ou d'optimisation sont exprimés comme des problèmes de sommes de sous-ensembles. Le plus classique, *somme de sous-ensembles*, demande, étant donné un ensemble E de N entiers et un entier $S \in \mathbb{N}$, s'il existe un sous-ensemble de E dont les éléments se somment à S . Ce problème est NP-complet, ce qui signifie que l'on ne connaît pas d'algorithme

permettant de le résoudre en un nombre d'opérations croissant polynomialement avec N . Il devient cependant de complexité polynomiale si l'on borne *a priori* la cardinalité du sous-ensemble. Il s'agit alors du problème suivant :

PROBLÈME k -SOMME : Étant donné un ensemble E de N entiers et un entier S , déterminer s'il existe s_1, \dots, s_k distincts dans E tels que $s_1 + \dots + s_k = S$.

Pour la mesure de la complexité dans ce problème, on fait l'hypothèse que l'addition et la comparaison d'entiers ont un coût unitaire, ce qui est réaliste pour des entiers stockés sur un nombre fixe de mots machines, comme ceux du type `int` de Java ou C.

De nombreuses questions de géométrie algorithmique se ramènent au problème k -somme. Par exemple, l'inclusion d'un polygone dans un autre à translation près se réduit à 4-somme. Plus simplement, étant donné N points du plan, décider si 3 d'entre eux sont alignés se réduit à 3-somme. Par ailleurs, 2-somme est utilisé semble-t-il comme question de test dans certains entretiens d'embauche.

2.1 Approche naïve

Question 9 Proposer un algorithme simple pour résoudre le problème 2-somme avec une complexité quadratique. \diamond

Question 10 En généralisant cette méthode, en déduire une première borne supérieure polynomiale sur la complexité du problème k -somme, pour $k \geq 2$. \diamond

2.2 Tri

Question 11 Proposer un algorithme permettant de résoudre le problème 2-somme sur une liste triée en complexité seulement linéaire. \diamond

Question 12 En déduire un algorithme de complexité quadratique pour 3-somme. \diamond

Cet algorithme, bien que très simple, est pratiquement le meilleur connu actuellement. Le problème de savoir si N^2 est l'ordre de grandeur d'une borne inférieure à la complexité de 3-somme est ouvert.

3 Diviser pour régner

La conception d'algorithmes efficaces repose souvent sur un principe simple : diviser pour régner. Cette idée fera l'objet du Cours 4. En voici quelques conséquences.

Question 13 Déduire de la question 11 un algorithme de complexité $O(N \log N)$ pour 2-somme. \diamond

Cette complexité $O(N \log N)$ est également une borne inférieure pour ce problème. Notre algorithme est donc optimal !

Question 14 (Plus difficile). Proposer un algorithme de complexité $O(N^2 \log N)$ pour 4-somme. (On pourra utiliser des tableaux auxiliaires). \diamond

La même idée montre que l'on peut obtenir un algorithme résolvant k -somme lorsque k est pair en complexité $O(N^{k/2} \log N)$, et une variante en $O(N^{(k+1)/2})$ lorsque k est impair. On est loin de la méthode naïve de la question 10 !

Références

- [1] J. Kleinberg et E. Tardős. *Algorithm Design*. Pearson, 2006.
- [2] D. E. Knuth. *Mariages stables*. Presses de l'université de Montréal, 1976.

- [3] A. E. Roth et J. H. Vande Vate. Random paths to stability in two-sided matching. *Econometrica*, 1990.
- [4] Anka Gajentaan et Mark H. Overmars. [On a class of \$O\(n^2\)\$ problems in computational geometry](#). *Computational Geometry. Theory and Applications*, 5(3):165–185, 1995. ISSN 0925-7721.
- [5] Nir Ailon et Bernard Chazelle. [Lower bounds for linear degeneracy testing](#). *Journal of the ACM*, 52(2):157–171, 2005. ISSN 0004-5411.
- [6] Jeff Erickson. [Lower bounds for linear satisfiability problems](#). *Chicago Journal of Theoretical Computer Science*, pages Article 8, 28 pp. (electronic), 1999. ISSN 1073-0486.