Algorithmique II
Année 2006-2007

Florent Bouchez
Jean-François Pineau

TD n^{o}9
Bzzz, bzzz, bzzz

Question 0.1. Rappel

Il y a n gentilles petites abeilles et m jolies petites fleurs. Chaque abeille a a peut être soit très attirée par une fleur soit pas du tout. Cet état vaut 0 ou 1, a été mesuré par un éromètre et est noté $\heartsuit(a, f)$. On suppose qu'aucune abeille n'est trop difficile pour n'aimer aucune jolie fleur, et qu'aucune fleur n'est trop repoussante au point de n'attirer aucune abeille. Comment maximiser l'amour cosmique entre les fleurs et les abeilles ?

Exercice 1 Prouver le théorème suivant :

Théorème 1 *(König – Egerváry) Dans un monde merveilleux d'abeilles et de fleurs, l'amour cosmique maximal est le coût d'une couverture minimale des relations abeille/fleur possibles par une sélection d'abeilles et de fleurs.*

Exercice 2 Prouver le théorème suivant :

Théorème 2 *(Hall) Montrer que toutes les gentilles petites abeilles peuvent trouver une fleur pour elle si et seulement si pour tout sous-ensemble A d'abeilles, l'ensemble* $N(A) = \{f \in \textit{jolies fleurs} \mid \exists a \in A \mid \heartsuit(a, f) = 1\}$ *est au moins aussi grand que A.*

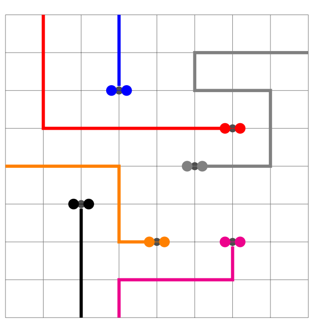
Exercice 3 Applications de Hall :

Question 3.1. *System of Distinct Representatives :*

Soit $(X_1, \ldots, X_n)$ une famille de sous-ensembles d'un ensemble A. Un SDR est une famille $(x_1, \ldots, x_n)$ d'éléments de A telle que pour tout i, $x_i \in X_i$ et pour tout $j \neq i$, $x_i \neq x_j$. Quelle est la condition d'existence d'un SDR ?

Question 3.2. Prouver que pour $k > 0$, si chaque abeille aime exactement k fleurs, et que chaque fleur est aimée par exactement k abeilles, alors il est toujours possible que chaque abeille trouve une fleur et que chaque fleur trouve une abeille.

Exercice 4 Tron v0.7



Des concurents sans pitié s'affrontent dans une arêne représentée par une grille. Chaque *lightcycle* part d'un point situé au bord de la grille, et se déplace sur les arêtes de la grille en laissant derrière lui un mur. Le but est évidemment d'éviter de se prendre le mur d'un collègue (ou même le sien) dans la tronche. Les coins aussi sont dangereux évidemment. On suppose qu'on connaît uniquement la position actuelle des *lightcycles* et pas où sont les murs.
Comment déterminer si cette position est valide ou non (i.e. vérifier que les joueurs ont effectivement pu arriver à leurs positions respectives en respectant les règles) ?

Exercice 5 Cache d'instructions *direct-mapped*.

On a une boîte très très longue et c couleurs différentes. La boîte est coloriée avec les couleurs : chaque unité de longueur a une couleur donnée, et l'ordre des couleurs est toujours le même, si bien que l'on a un coloriage périodique de période c unités de longueur.

Dans notre boîte on veut mettre n gros serpents. Les gros serpents sont de taille entière en unités de longueur, et sont aussi coloriés, de la même manière que la boîte. On connaît la taille de tous les serpents, et aussi la couleur de leur queue (ce qui fait qu'on sait exactement comment ils sont coloriés).

Le but est de rentrer tous les serpents dans la boîte, sachant que dans une partie de la boîte on ne peut mettre qu'un serpent (ils sont gros), et en plus, chaque partie du corps d'un serpent doit être de la même couleur que la partie de la boîte qui l'entoure.

Exemple : c = {rouge,vert,bleu}, et trois serpents *vbr*, *b*, et *br*. Alors une solution possible est :

b r	v b r		b

avec un trou entre le deuxième et le troisième serpent.

Question 5.1. Proposez une méthode qui marche et prouvez que la longueur de boîte utilisée n'est pas optimale.

On suppose qu'on a le droit d'utiliser plusieurs boîtes. Les boîtes peuvent commencer à n'importe qu'elle couleur mais toutes doivent avoir une taille multiple de c. On veut minimiser la somme des longueurs des boîtes utilisées.

Question 5.2. Modélisez à l'aide de graphes bipartis et prouvez qu'une méthode gloutonne renvoie l'optimal.

Question 5.3. Modifiez la solution trouvée à la question précédente pour obtenir une solution de même coût qui est minimale en nombre de boîtes utilisées (on ne prouvera pas la minimalité).

Question 5.4. La solution obtenue à la question précédente a un coût ¢ minimal. S'il restait au moins deux boîtes, montrez comment la modifier pour obtenir une solution avec une seule boîte. Quelle est la longueur minimale de cette boîte ?

On revient au cas initial : on n'a qu'une boîte et sa taille n'est pas forcément multiple de c.

Question 5.5. Utilisez l'algorithme de la question précédente pour obtenir une solution optimale.