Master Informatique 1ère année - Université de Lorraine

Algorithmique et Complexité

Projet

Ce projet doit être réalisé par groupes de 3 étudiants.

Il devra être remis sur arche au plus tard le 16 janvier 2017 sous la forme d'une archive `.zip` ou `.gz` contenant les programmes demandés et un rapport pdf (incluant les réponses aux questions).

Attention : un logiciel de détection de plagiat sera utilisé, et ses indications seront prises en compte dans la notation.

1 Prise en main du problème : motifs interdits

On s'intéresse dans cette partie au problème suivant. On suppose avoir n variables $x_1 \dots x_n$, et on veut affecter une couleur à chacune des variables, à choisir entre rouge, vert, et bleu, de sorte à respecter certaines contraintes. On utilisera les abréviations R, V, B pour les couleurs.

Une *contrainte* est une liste de couples (variable, couleur).
Par exemple, $C = [(x_1, R), (x_3, V), (x_4, B)]$ est une contrainte.

Une assignation ϕ des variables est une fonction qui associe à chaque variable une couleur. Par exemple $\phi : x_1 \mapsto B, x_2 \mapsto V, x_3 \mapsto V, x_4 \mapsto B$ est une assignation.

On dira qu'une assignation ϕ satisfait une contrainte C si l'assignation est incompatible avec la contrainte, c'est à dire s'il existe une variable qui prend une valeur différente dans l'assignation et dans la contrainte. Dans l'exemple ci-dessus, ϕ satisfait la contrainte C (puisque x_1 n'a pas la même valeur dans l'assignation et dans la contrainte).

Etant donné un ensemble de contraintes $\mathcal{C}$, on dira que ϕ satisfait $\mathcal{C}$ si ϕ est incompatible avec chacune des contraintes de l'ensemble $\mathcal{C}$. On dit que $\mathcal{C}$ est satisfiable s'il existe une assignation qui satisfait $\mathcal{C}$.

Q 1) Montrer que $\mathcal{C}$ ci-dessous est satisfiable. Pour cela, trouver toutes les assignations qui satisfont $\mathcal{C}$.

$$\mathcal{C} = \{[(x_1, R)], [(x_1, B), (x_2, B)], [(x_1, V), (x_2, R)]\}$$

Q 2) Montrer que $\mathcal{C}$ ci-dessous n'est pas satisfiable.

$$\mathcal{C} = \left\{ \begin{array}{l} [(x_1,R),(x_2,R)]\ ,\ [(x_1,V),(x_2,V)]\ ,\ [(x_1,B),(x_2,B)] \\ [(x_1,R),(x_3,R)]\ ,\ [(x_1,V),(x_3,V)]\ ,\ [(x_1,B),(x_3,B)] \\ [(x_1,R),(x_4,R)]\ ,\ [(x_1,V),(x_4,V)]\ ,\ [(x_1,B),(x_4,B)] \\ [(x_2,R),(x_3,R)]\ ,\ [(x_2,V),(x_3,V)]\ ,\ [(x_2,B),(x_3,B)] \\ [(x_2,R),(x_4,R)]\ ,\ [(x_2,V),(x_4,V)]\ ,\ [(x_2,B),(x_4,B)] \\ [(x_3,R),(x_4,R)]\ ,\ [(x_3,V),(x_4,V)]\ ,\ [(x_3,B),(x_4,B)] \end{array} \right\}$$

2 Algorithme randomisé

Le but de cette partie est de fournir et d'implémenter un algorithme randomisé qui décide s'il existe une assignation qui satisfait un ensemble de contraintes.

Dans cette partie, on regarde uniquement les problèmes où les contraintes sont binaires (concernent deux variables) ou unaires (concernent une seule variable). On note $(3,2) - SSS$ le problème correspondant.

L'algorithme proposé fonctionne de la manière suivante : il supprime variable après variable jusqu'à ce qu'il aboutisse à une contradiction, ou qu'il n'y ait plus de variables.

Il fonctionne suivant 4 cas :

1. Soit il y a une variable x qui apparaît dans 3 contraintes unaires, $[(x,R)]$, $[(x,V)]$ et $[(x,B)]$. Auquel cas l'ensemble de contraintes n'est pas satisfiable
2. Soit il existe une variable x qui apparaît dans 2 contraintes unaires, par exemple $[(x,R)]$, $[(x,B)]$. Auquel cas on peut éliminer la variable x.
3. Soit il existe une variable x qui apparaît dans une seule contrainte unaire, par exemple $[(x,R)]$. Auquel cas on peut éliminer x de la façon suivante.
   — Eliminer toutes les contraintes de la forme $[(x,R),(y,\cdot)]$
   — Pour tout couple de contraintes $[(x,V),(y,b)]$ et $[(x,B),(z,c)]$, ajouter la contrainte $[(y,b),(z,c)]$
4. Sinon, il n'y a que des contraintes binaires.
   — On prend la première, disons $[(x,R),(y,B)]$.
   — Choisir au hasard (chacune avec probabilité 1/4) une des quatre possibilités suivantes :
      — Ajouter $[(x,R)]$ et $[(y,R)]$
      — Ajouter $[(x,R)]$ et $[(y,V)]$
      — Ajouter $[(x,B)]$ et $[(y,B)]$
      — Ajouter $[(x,V)]$ et $[(y,B)]$
   — Supprimer x et y en utilisant les 3 premiers cas.

Q 3) Expliquer plus précisément ce qu'il faut faire dans le deuxième cas

Q 4) Expliquer pourquoi le troisième cas est correct.

Q 5) Dans le dernier cas, on note $\mathcal{C}$ l'ensemble de contraintes avant l'ajout, et $\mathcal{C}'$ après l'ajout. Montrer que :

— Si $\mathcal{C}$ n'est pas satisfiable alors $\mathcal{C}'$ est satisfiable

— Si $\mathcal{C}$ est satisfiable alors $\mathcal{C}'$ est satisfiable avec probabilité $1/2$.

On déduit des questions précédentes que l'algorithme complet (qui part de n variables jusqu'à supprimer toutes les variables) est correct avec probabilité au moins $1/2^{n/2}$. En le répétant $10 \times 2^{n/2}$ fois, on obtient donc un algorithme qui est correct 99.995% du temps.

Q 6) Implémenter l'algorithme. Utiliser-le pour tester si les 10 graphes fournis sont coloriables.

Note pour l'implémentation : Il ne faut pas représenter les ensembles de contraintes par des listes : en effet, ajouter des contraintes (ce qu'on fait dans les cas 3 et 4) nécessiterait de parcourir la liste pour vérifier si on n'est pas en train d'ajouter une contrainte qui existe déjà.

En supposant qu'on représente les 3 couleurs par $0, 1$ et 2, une bonne solution pour l'implémentation est de représenter l'ensemble de contraintes par deux tableaux.

— Un tableau de booléens $contrainte[n][n][3][3]$. Par exemple $contrainte[3][4][0][2]$ sera égal à vrai si la contrainte $[(x_3, 0), (x_4, 2)]$ est présente.

— Un tableau de booléens $unaire[n][3]$. Par exemple $unaire[5][1]$ sera égal à vrai si la contrainte $[(x_5, 1)]$ est présente.

3 Motifs interdits pondérés

On considère maintenant une variante du problème des motifs interdits, où chaque contrainte C_i de l'ensemble de contraintes $\mathcal{C}$ est munie d'un poids strictement positif w_i. Le but est désormais de trouver une assignation des variables qui maximise le poids cumulé des contraintes satisfaites. Plus précisément, on s'intéresse maintenant au problème $maxW-SSS$ qui cherche le poids cumulé maximal des contraintes pouvant être satisfaites par une même assignation des variables.

3.1 NP-complétude

Q 7) Formuler précisément le problème de décision $W - SSS$ associé à ce problème d'optimisation $maxW - SSS$.

Montrer que le problème $(3, 2)-SSS$ se ramène à $W-SSS$ via une réduction polynômiale.

Q 8) Montrer également que les problèmes $2-SAT$ et $3-COLOR$ se ramènent à $(3, 2)-SSS$ via des réductions polynômiales.

Q 9) Conclure sur la NP-complétude de $(3, 2) - SSS$ et de $W - SSS$. On n'oubliera pas de traiter l'appartenance à NP.

3.2 Algorithme glouton

Q 10) Proposer un algorithme glouton pour le problème $maxW - SSS$:

— expliciter et justifier le tri des données

— décrire précisément l'algorithme
— donner sa complexité

Q 11) Cet algorithme glouton est-il optimal ? Justifier.

Q 12) Implémenter l'algorithme, en veillant à fournir suffisamment d'indications sur le déroulement de son exécution. Préciser quelques ensembles de contraintes utilisés pour vos tests. Attention : désormais les contraintes ne sont plus seulement unaires ou binaires, donc la représentation utilisée précédemment n'est plus forcément adaptée.