Projet du parcours MI

Voyage dans le monde des contraintes

11 juin 2012

Coloration de graphes, ajustement d'intervalles et emploi du temps

L'objectif de ce projet est de permettre à l'élève ingénieur de se familiariser avec la problématique de la résolution des contraintes et de mettre ainsi en œuvre ses connaissances dans le domaine des graphes et de l'algorithmique. Nous étudions dans ce projet deux algorithmes de résolution de contraintes qui sont complètement indépendants. Il vous est demandé **d'effectuer l'étude théorique de chacun et d'en choisir ensuite un des deux pour la partie implémentation**. Le premier algorithme utilise des notions de graphes bipartis et de coloration d'arêtes. Cette coloration va nous permettre de résoudre une version simple du problème d'emploi du temps. Le deuxième algorithme traite des problèmes d'ajustement d'intervalles dans le but de réduire le domaine de la recherche de la solution et donc d'optimiser la résolution des contraintes.

1 Coloration de graphes : rappel

Il existe deux types de coloration d'un graphe : la coloration des sommets et la coloration des arêtes.

Nous avons vu (cours théorie de graphes) que la coloration des sommets d'un graphe consiste à trouver le plus petit nombre de couleurs tel qu'il n'existe pas de sommets adjacents de la même couleur et nous avons montré comment résoudre certains problèmes de contraintes avec ce type de coloration.

Nous nous intéressons dans le cadre de ce projet à la coloration des arêtes de graphes et plus particulièrement à l'application de cette coloration sur des graphes bipartites. Nous appliquons les algorithmes proposés pour la résolution d'un problème simple d'emploi du temps.

Nous introduisons dans la suite quelques notions de coloration d'arêtes dans un graphe, comme l'indice chromatique et le couplage.

Étant donné un graphe G et un entier k, on appelle k-coloration des arêtes de G toute application dans un ensemble de couleurs telle que deux arêtes ayant une extrémité commune soient associées à deux couleurs différents.

De même **l'indice chromatique** du graphe G appelé $q(G)$ est le plus petit entier k, tel qu'il existe une k-coloration de G.

Pour chaque couleur, l'ensemble d'arêtes ayant cette couleur est appelé **couplage** (dans lequel deux arêtes quelconque n'ont pas d'extremités communes). Une coloration d'un graphe G est une partition de l'ensemble d'arêtes en couplages.

1.1 Résultats et théorèmes

On vérifie que :

$$\Delta \leq q(G) \leq m$$

où Δ est le degré maximum dans le graphe et m étant le nombre d'arêtes.

THÉORÈME 1 Si G est un graphe simple, alors $g(G)$ est égal à Δ ou $\Delta + 1$.

THÉORÈME 2 Si G est biparti, alors $g(G)$ est égal à Δ.

Noter que dans un graphe biparti, il est possible de trouver un algorithme polynomial pour une $\Delta-$coloration.

1.2 Le problème de l'emploi du temps

Nous décrivons et nous modélisons dans cette section un problème d'emploi du temps simple. Soit X un ensemble de professeurs, soit Y l'ensemble de classes et soit E l'ensemble de cours à donner *de durée d'une heure chacun*, nous construisons le graphe biparti $G = (X, Y, E)$ modélisant ce problème.
Un emploi du temps correspond à une partition de l'ensemble des arêtes en couplages.

La question qu'on cherche à résoudre est la suivante : quel est le plus petit nombre d'heures en lequel il est possible d'établir l'emploi du temps ?

1.3 Contrainte de salles

Nous ajoutons à ce qui précède, une contrainte sur le nombre de salles disponibles appelé s : s étant le nombre de salles disponible. Soit m le nombre total d'heures de cours : $m =\mid E \mid$, soit k le plus petit nombre d'heures pour lequel il est possible d'établir un emploi du temps. Notons que :

$$k \geq max(\Delta, \left\lceil \frac{m}{s} \right\rceil)$$

Nous cherchons à proposer un emploi du temps faisable en k heures.

Lemme 1 Soient $G = (X, Y, E)$ un graphe biparti, k un entier $\geq \Delta$. Il existe une partition de E en k couplages ayant à une unité près le même nombre d'éléments.

Preuve Par construction, nous équilibrons les couleurs.

- **Répéter**
  1. Choisir les deux couplages C_i,C_j ayant la différence maximale de longueurs.
  2. Procéder à l'équilibrage en échangeant quand nécessaire, les couleurs dans les chaînes de longueur impaire.
- **jusqu'à** l'équilibrage complet du graphe.

Remarque Les k couplages ont chacun $\left\lceil \frac{m}{k} \right\rceil$ ou $\left\lfloor \frac{m}{k} \right\rfloor$ éléments.

1.4 Étude d'un exemple

Soient x_1, x_2, x_3, x_4 4 professeurs, soient y_1, y_2, y_3, y_4, y_5 5 classes, soit le tableau de contraintes suivant contenant le nombre d'heures de cours qu'un professeur x_i doit effectuer pour une classe y_j :

	y1	y2	y3	y4	y5
x1	1	2	0	0	0
x2	1	1	1	0	0
x3	0	1	1	1	1
x4	0	0	0	1	2

Remarquer que $m = 13$, $\Delta = 4$, et qu'un emploi du temps en 4 heures est possible. Ceci suppose qu'au moins $\left\lceil \frac{m}{4} \right\rceil = 4$ salles disponibles.

Supposons qu'il y ait seulement 3 salles disponibles : $max(\Delta, \left\lceil \frac{m}{s} \right\rceil) = max(4, 5) = 5$. Le tableau **??** donné en annexe montre le déroulement de l'algorithme du ré-équilibrage des couleurs.

L'emploi du temps trouvé après déroulement de l'algorithme sera formulé par le tableau **??**.

heure	1	2	3	4	5
x1	-	-	y2	y2	y1
x2	-	y3	y1	-	y2
x3	y3	y2	y5	y4	-
x4	y4	y5	-	y5	-

TABLE 1 – emploi du temps résultant

1.5 Travail théorique : problème à résoudre

On considère l'emploi du temps suivant en 6 heures, avec 4 professeurs : $p_1, \ldots, p_4$ et 5 classes : $c_1, \ldots, c_5$ (voir tableau **??**).

Question 1 Combien cet emploi du temps nécessite-t-il de salles ?

heures	1	2	3	4	5	6
p_1	$c1$	$c3$	$c5$	$-$	$c4$	$c5$
p_2	$c2$	$c1$	$c1$	$c3$	$-$	$-$
p_3	$-$	$c4$	$-$	$c2$	$-$	$c1$
p_4	$c4$	$c5$	$c2$	$c1$	$c3$	$c3$

TABLE 2 – Emploi du temps proposé

Question 2 Montrer sans refaire cet emploi du temps, qu'il est possible d'établir un emploi du temps pour le même ensemble de cours et toujours en 6 heures, mais avec une salle en moins.

Question 3 Établir un emploi du temps répondant à la question précédente avec en outre la condition que le professeur p_2 reste libre dans les deux dernières heures.

Question 4 Toujours pour le même ensemble de cours, combien faudrait-il d'heures au minimum pour établir un emploi du temps avec une salle de moins encore (que la question précédente) ?

Question 5 Appliquer l'algorithme de ré-équilibrage des couplages pour établir l'emploi du temps demandé dans la question précédente.

1.6 Travail pratique

Écrire un programme qui permet d'établir un emploi du temps avec un minimum de nombre d'heures à partir d'un tableau de contraintes donné (comme par exemple le tableau **??**) et du nombre de salle disponibles. Tester votre programme sur les exemples proposés dans ce poly et sur d'autres jeux de tests que vous proposez.

2 Problème d'ordonnancement et contraintes avec intervalles

Il s'agit de traiter un ensemble de contraintes pour affecter un ensemble de variables aux valeurs adéquates. Nous étudions dans cette section une contrainte globale très connue appelée la contrainte $\text{all}_different(L) o \text{L}estunelistedevariables, chaquevariableestdfiniesurundomainedevaleurs.$

Exemple d'un problème d'ordonnancement avec des contraintes exprimées par des intervalles Soit le problème d'ordonnancement suivant : nous souhaitons utiliser une salle pour une suite d'exposés, un exposé dure une heure. Le tableau **??** montre les contraintes des intervenants exprimées par intervalles. Par exemple Jean peut commencer son exposé à 3h, à 4h,à 5h ou à 6h.

Intervenant	début	fin
Jean	3	6
Marie	3	4
Grég	2	5
Sue	2	4
Paul	3	4
Hélèna	1	6

TABLE 3 – Exemple des contraintes exprimées par des intervalles

Nous exprimons ces contraintes sous forme d'un ensemble de variables chacune ayant comme domaine un intervalle : $J \in [3,6]$, $M \in [3,4]$, $G \in [3,5]$, $S \in [2,4]$, $P \in [3,4]$, $H \in [1,6]$.

La contrainte globale qu'on cherche à résoudre est : $\text{all}_different$([J,M,G,S,P,H]).

2.1 Notions théoriques

Nous introduisons ici quelques notions théoriques nécessaires pour la suite.

Soit K un ensemble de variables, on définit $dom(K) = \bigcup_{x_i \in K} D_i$ où D_i est le domaine (un intervalle de valeurs entières) de définition de X_i.

Théorème 3 : La contrainte $\text{all}_different(\text{x}_1, \ldots, x_n)$ sur les variables $X_1, \ldots, X_n$ de domaines $D_1, \ldots, D_n$ a une solution ssi il n'existe aucun sous-ensemble $K \subseteq \{X_1, \ldots, X_n\}$ tel que $|K| > |dom(K)|$.

Remarquer que s'il existe un ensemble K tel que $|K| = |dom(K)|$, les valeurs de $dom(K)$ sont utilisées pour les variables de K.

Par conclusion on supprime ses valeurs des variables qui n'appartiennent pas à K. Exemple : $K = \{M, S, P\}$, $K = \{M, P\}$

Intervalle de Hall : Étant données les variables $\{X_1, \ldots, X_n\}$ de domaines $D_1, \ldots, D_n$ et un intervalle I. Soit $vars(I) = \{X_i | D_i \subseteq I\}$. L'intervalle I est dit de Hall ssi $|I| = |vars(I)|$.

La contrainte all$_different(X_1, \ldots, X_n)estdite$**consistantedebornes***parrapport*D$_1, \ldots, D_n$ ssi :

- Pour chaque intervalle I on a $|vars(I)| \leq |I|$.
- Pour chaque intervalle de Hall J et chaque variable X_i, on a : soit $D_i \subseteq J$, soit $\{minD_i, maxD_i\} \bigcap J$ ϕ

2.2 Ajustement des intervalles

Pour pouvoir retrouver la solution rapidement et d'une façon plus optimisée, nous devons rendre la contrainte all_different consistante de bornes en réduisant les bornes des intervalles. Deux étapes sont nécessaires : la mise à jour des bornes gauches, ensuite celles de droites.

Nous présentons ici la méthode de mise à jour des bornes gauches. Les étapes de l'algorithme sont les suivantes :

1. Tri des variables par ordre croissant des bornes supérieures :
   - dans l'exemple (voir le tableau **??**) nous obtenons l'ordre : Marie, Sue, Paul, Grég, Jean, Hélèna.
2. Détermination des intervalles de Hall.
3. Modification des bornes gauches.

Nous présentons dans la suite les trois procédures nécessaires pour la réalisation de cet algorithme :

Procédure $maj\_min(x_1, \ldots, x_n)$
- Trier(X)
- Pour $i = 1$ à n faire
  - $min[i] = min(x[i])$
  - $max[i] = max(x[i])$
- Pour $i = 1$ à n faire
  - $inserer(i)$

Procedure $inserer(i)$ /* cette procédure traite la variable numéro i*/
/* u est une variable globale qui permet de détecter des intervalles de Hall */
- $u[i] = min[i]$
- Pour $j = 1$ à $i - 1$ faire
  - **SI** $min[j] < min[i]$
    - $u[j] + +$
    - **SI** $u[j] > max[i]$ echec
    - **SI** $u[j] = max[i]$ IncrMin(min[j],max[j],i)
  - **SINON** u[i]++
- **SI** $u[i] > max[i]$ echec
- **SI** $u[i] = max[i]$ IncrMin(min[i],max[i],i)

Procédure IncrMin(a,b,i) /* cette procédure permet de modifier la borne gauche des variables qui ne sont pas concernées par l'intervalle de Hall [a,b] */

- Pour $j = i + 1$ à n faire
  - **SI** $min[j] \geq a$
    - $min[j] \leftarrow b + 1$

2.3 Travail théorique

Question 1 Faire tourner l'algorithme proposé pour ajuster les bornes gauches des intervalles sur l'exemple proposé dans le tableau **??**.

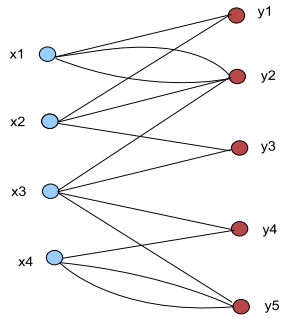
Question 2 Proposer un algorithme similaire qui traite les bornes droites des intervalles.

Question 3 Proposer un algorithme qui rend une contrainte de type all_different, avec domaines représentés par des intervalles, consistance de bornes.

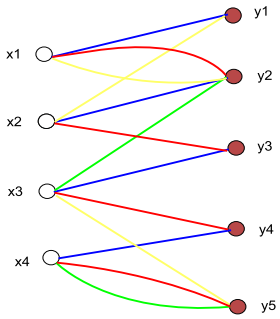
2.4 Travail pratique

Écrire un programme qui a comme objectif de rendre la contrainte allDiff consistante de bornes et d'afficher à la fin les intervalles résultants. Ce programme peut être appliqué directement à l'exemple proposé ici (tableau **??**). Les élèves qui réaliseront un programme général obtiendront un bonus.

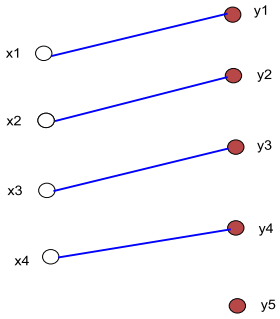
Le graphe complet



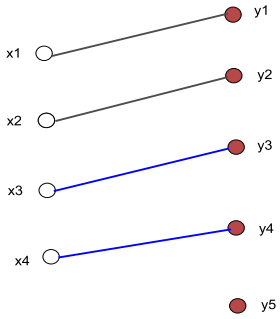
5 couplages : C1 : bleu, C2 : rouge, C3 : Jaune, C4 : vert, C5 : vide.



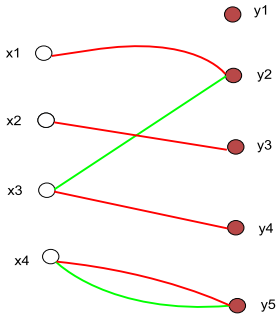
G : C1 & C5 :



Échange entre C1 & C5



G : C2 & C4



Echange entre C2 & C4 : graphe équilibré :

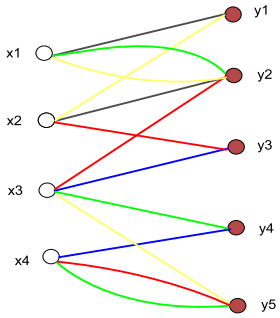


TABLE 4 – Étapes du ré-équilibrage des couplages