Intelligence Artificielle : Logique et Contraintes - TD n° 5

TP : Premiers pas avec OPL

Pour cette séance et les séances suivants de ce cours, nous allons modéliser et résoudre différents problèmes en utilisant le langage `OPL` de la suite *IBM CPLEX Optimization Studio*.

Pour organiser le code que vous allez écrire pour ce tp et ceux des prochaines séances, nous vous suggérons d'organiser votre compte utilisateur en utilisant un répertoire pour ce cours (*e.g.* `cours_ialc`) contenant un dossier `opl` dans lequel vous créerez **un dossier par problème considéré**.

Pour chaque problème, vous mettrez dans le dossier correspondant le (ou les) fichier(s) correspondant au modèle(s) à réaliser (i.e. les fichiers suffixés par `.mod` et si nécessaire les fichiers correspondant aux données).

Pour cette séance, vous pouvez télécharger sur la page du cours une archive toute prête, reprenant cette structure, que n'aurez qu'à décompresser et compléter :

`https://ecampus.paris-saclay.fr/course/view.php?id=40690` (section TDs / TPs)

Dans cette archive, vous constaterez que le répertoire `opl` contient deux dossiers correspondant aux deux exercices ci-dessous, ainsi qu'un répertoire `shared`, dans lequel vous pourrez mettre les portions de code qui peuvent être réutilisés d'un problème à l'autre (typiquement des variantes de blocs de contrôle de flux).

L'objectif de cette séance est de se familiariser avec les constructions de base du langage `OPL` pour des problèmes relativement simples.

I) Jeux de billes...

A partir d'un énoncé de problème que nous formalisons (indépendamment du langage du solveur utilisé), nous explorons différentes manières d'encoder ce modèle avec OPL.

Exercice 1 (Formalisation)

Quatre enfants, Anne, Bernard, Claudine et Denis, ayant tous un âge différent entre 4 et 7 ans, jouent chacun à différents endroits avec des billes de couleur bleue, jaune, noire ou rouge.

1. Denis joue dans le parc et n'a pas 4 ans, contrairement à l'enfant qui a des billes bleues.
2. La fille de 6 ans a des billes jaunes.
3. L'enfant qui joue avec des billes noires est plus âgé que l'enfant qui joue dans le jardin mais plus jeune que Anne.
4. Anne, qui joue dans sa chambre, a 1 an de plus que l'enfant qui joue dans le salon.

1. Formalisez l'énoncé précédent sous forme d'un CSP P (indépendamment de la syntaxe du solveur utilisé). Vous pouvez répondre à cette question en complétant le fichier texte `billes.txt` dans lequel :
   - vous décrirez les variables, leurs domaines respectifs et préciserez le sens de ces variables,
   - vous recopierez chaque (portion de) phrase pertinente de l'énoncé en la faisant suivre par la(les) contrainte(s) qui la modélise(nt).

Exercice 2 (Codage du problèmes des billes en OPL)

1. En partant du modèle de l'exercice précédent, compléter le fichier `billes1.mod` pour décrire ce modèle dans la syntaxe du langage OPL. On souhaite dans un premier temps une solution où les noms des variables de décision correspondent à celles du modèle précédent.
   Ecrire un bloc de script permettant de paramétrer le solveur avec la stratégie `"DepthFirst"`.
   Ecrire un bloc de script de post-traitement permettant d'afficher la solution.
2. Le code précédent à l'avantage de coller au plus près à l'énoncé d'origine mais n'est pas forcément très pratique car il est complètement spécifique au problème considéré. Si on veut rajouter une personne, un trait de caractère, il va falloir apporter des modifications dans plusieurs parties du modèle.
   On se propose donc d'écrire un second modèle (compléter le fichier `billes2.mod`) utilisant des ensembles (ou des range) et permettant de regrouper les variables dans un ou plusieurs tableaux, indexés de façon adéquate. L'objectif est de faciliter l'énoncé de certaines contraintes ainsi que l'affichage de la solution, dans le bloc de post-traitement.
3. Copiez le modèle précédent dans le fichier `billes2_all.mod` et rajouter une section de contrôle de flux permettant d'énumérer toutes les solutions.
   Pour faciliter la lecture du résultat, rajouter un compteur permettant de numéroter les solutions et de séparer l'affichage de chaque solution de façon claire.
4. Le bloc de contrôle de flux que vous venez d'écrire étant très générique, il pourrait être utilisé dans des modèles décrivant d'autres problèmes. Isolez le dans un fichier externe `"allSolutions.mod"` (dans le répertoire `shared`). Dupliquez le contenu du fichier précédent dans un fichier `billes2_all2.mod`, dans lequel vous remplacerez simplement le bloc de contrôle de flux par une directive permettant d'inclure (avec un `include`) le fichier `"allSolutions.mod"`

II) Formalisation de problèmes paramétrés

On souhaite à présent modéliser des problème dont le nombre de variables dépend de la valeur d'un paramètre donné.

Exercice 3 Le problème des Carrés Latins en OPL

Un *carré latin* d'ordre n est une matrice $n * n$, dont tous les éléments figurent parmi n symboles distincts (habituellement les entiers $1, 2, \ldots, n$) et sont disposés de telle façon que chaque symbole apparaisse une fois et une seule sur chaque ligne et chaque colonne.

1. Formaliser ce problème en terme de problème de satisfaction de contraintes.
2. Ecrire un modèle OPL permettant de construire un carré latin d'ordre n. Vous ferez en sorte d'afficher uniquement la 1ère solution trouvée et de compter nombre total de solutions. Testez le résultat pour différentes valeurs de de n.