Réalisation d'un mini solveur de CSP binaires

MPRO, PPC

L'objectif du projet est d'implémenter un solveur générique de CSP où les contraintes sont binaires et les variables entières. Les domaines des variables seront finis. Il vous faudra modéliser un tel CSP, implémenter une ou plusieurs méthodes de consistance ainsi qu'un moteur de résolution. Vous testerez votre solveur sur de petits problèmes.

1 Travail minimal

1.1 Modélisation

Tout d'abord, il faut écrire un modèle pour ce type de CSP. Nous vous rappelons qu'un problème est défini par un ensemble de variables, un domaine pour chaque variable et un ensemble de contraintes. Les contraintes sont binaires, elles sont donc définies par deux variables et un ensemble de tuples qui satisfont la contrainte. Un tuple est défini par deux valeurs ordonnées.

1.2 Wrapper

Il pourra être intéressant de faire un *wrapper* qui permet d'ajouter facilement des contraintes très courantes à vos modèles (alldiff, somme pondérée, etc.).

1.3 Branchement

Implémenter un algorithme de backtrack. Il faudra pouvoir changer facilement le choix des heuristiques utilisées (choix des variables, choix des valeurs).

1.4 Consistance

Il faudra intégrer un algorithme d'arc-consistance (AC3 ou AC4) et le forward-ckecking. Ces derniers devront être facilement activés ou désactivés dans la recherche arborescente selon les choix de l'utilisateur.

1.5 Benchmarks

1.5.1 n-Reines

Vous commencerez par tester votre code sur le problème des n-Reines. Chaque instance est réduite à un entier n qui définit la taille du côté de la grille.

1.5.2 colorabilité

Vous trouverez sur cette page des instances de graphes qu'il s'agit de colorer avec le minimum de couleurs différentes :

`http://mat.gsia.cmu.edu/COLOR/instances.html`

Attention c'est un problème d'optimisation ici ! Vous pouvez regarder quelles instances vous arrivez à résoudre.

1.5.3 autres problèmes

Ce site recense beaucoup de problèmes de type CSP :

`http://www.csplib.org/`

Vous pouvez y piocher des problèmes qui vous semblent intéressants et vous benchmarker par rapport aux résultats existants.

2 Approfondissement

La réalisation du travail minimal décrit plus haut devrait avoir soulevé des questions, des interrogations. Je voudrais que vous choisissiez un point qui vous semblera intéressant et que vous le développiez. Tester une idée, comprendre un phénomène, vous êtes libres de jouer avec le solveur que vous avez fabriqué. Voici quelques exemples de pistes possibles :

- étude des symétries : dans la plupart des modèles, il y a des symétries (de valeur, de variable, des deux). Comment les limiter ? Quel impact sur le temps de calcul ?
- influence des heuristiques de branchement : comment la mesurer ? étude expérimentale, conclusion.
- hybridation avec un autre solveur : quel intérêt ? qui fait quoi ? expérimentations.
- étude plus fine de l'impact de la consistance : différence entre les stratégies (MAC, FC, MAC à la racine seulement, etc.), quelle réduction sur la taille de l'arbre parcouru ? quelle réduction sur le temps de calcul ?
- étude de l'impact de la modélisation : prendre un problème assez difficile et réaliser différentes modélisation afin de voir l'impact sur les temps de résolution.
- implémentation de technique de look-back et utilisation expérimentale.

3 Rendus

Lors de la dernière séance, vous ferez une soutenance et vous rendrez un rapport relativement court expliquant vos choix, les points forts et les points faibles de votre solveur, les résultats sur les benchmarks. Vous présenterez également la question que vous avez choisie d'approfondir et les conclusions associées.