TP1 : Benchmark de résolutions de Sudokus

Sujet 4 : Résolution par CSP avec AIMA

Peter Norvig :

Scientifique américain, chercheur en informatique né le 12 mai 1956.

Il est diplomé en mathématique appliquées de l’université de Brown, et en informatique de l’université de Berkeley.

Il a travaillé à la NASA, chez Sun Microsystems et travaille depuis 2001chez Google en tant que directeur de recherche chez Google. Il publie de nombreux article scientifique sur l’informatique et notamment dans le domaine de l’AI :

* Artificial Intelligence : A modern approach
* Paradigms of AI Programming : Case Studies in Common Lisp
* Verbmobil : A translation System for Face-to-Face Dialog

Résolution de sudokus par CSP (Problème de satisfaction de contraintes) :

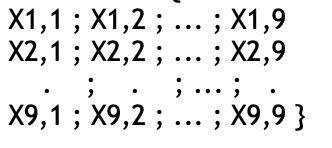
La programmation par contraintes est utilisée pour résoudre des problèmes combinatoires tel que les problèmes de planification et d’ordonnancement comme le sudoku.

Définition d’une contrainte : Elle est définit en 3 variables X, D, C

X : Ensemble de variables du problème

D : Ensemble des domaines variables

C : Ensemble des contraintes

Variable X : Les variables associées au Sudoku correspondent aux cases de la grille donc X vaut :

Variable D : correspond à l’ensemble des domaines de variables, soit l’ensemble de valeurs que peut prendre une variable. Ainsi D = [1,2,3,4,5,6,7,8,9]

Variable C : correspond à l’ensemble des contraintes. On cherche à avoir que toutes les variables présentes sur une même ligne soient différentes, que toutes les variables présente sur une colonne soit différentes et enfin que toutes les variables présentes dans un même bloc soient différentes. On modélise les contraintes de la façon suivante :

Pour les lignes :

allDifferent(x11, x12, x13, x14, x15, x16, x17, x18, x19),  
allDifferent(x21, x22, x23, x24, x25, x26, x27, x28, x29),  
...  
allDifferent(x91, x92, x93, x94, x95, x96, x97, x98, x99)

Pour les colonnes :

allDifferent(x11, x21, x31, x41, x51, x61, x71, x81, x91),  
allDifferent(x12, x22, x32, x42, x52, x62, x72, x82, x92),  
...  
allDifferent(x19, x29, x39, x49, x59, x69, x79, x89, x99)

Pour les blocs :

allDifferent(x11, x12, x13, x21, x22, x23, x31, x32, x33),  
allDifferent(x14, x15, x16, x24, x25, x26, x34, x35, x36),  
...  
allDifferent(x77, x78, x79, x87, x88, x89, x97, x98, x99)

Le sudoku fait partie des jeux à n’avoir qu’une seule et unique solution possible. Il n’existe pas de sudoku ayant plusieurs solutions. Le nombre de grille possible est de : 6,67.1021 !!

Résolution

Une solution à un CSP est d’affecter à chaque variable une valeur de son domaine de manière à satisfaire toutes les contraintes.

Pour cela le solveur de CSP permet de trouver une solution au CSP ou bien de prouver qu’aucune solution n’existe.

Librairie AIMA – Java et CSPs  :

AIMA signifie Artificial Intelligence : A Modern Approach.

C’est un projet crée et développé par Peter Norvig et Russell qui a été accepté pour le Google Summer of Code (GSoC) en 2016. Il est développé en plusieurs langages comme le python, le C-sharp, le lisp … et est d’ailleurs en cours de développement pour certains langages. D’ailleurs grâce à une repository git, chacun peut contribuer à faire avancer le projet.

Ci-joint les explications complètes du code :

<https://github.com/aimacode/aima-pseudocode/blob/master/aima3e-algorithms.pdf>

<https://github.com/aimacode/aima-pseudocode/blob/master/aima4e-algorithms.pdf>

Voir aussi site officiel de Peter Norvig sur l’AIMA : http://aima.cs.berkeley.edu/adoptions.html

IKVM : Port en #c de librairie Java :

IKVM.ney est une implémentation de Java pour Mono et microsoft.net Framework. Il contient les composants suivants :

* Une machine virtuelle Java implémentée dans .NET
* Une implémentation .NET des bibliothèques de classes Java
* Outils permettant l’interopérabilité Java et .NET.