Rapport TP1 - Sudoku

Alexis Charette
Francis de Ladurantaye

19 mars 2018

1 Formulation du problème

1.1 Notion d'état

Pour le problème du sudoku, un état correspond à l'ensemble des cases, chacune accompagnée de sa valeur. Comme valeur, une case peut contenir un chiffre dont nous sommes certains (cases initiales), un chiffre dont nous sommes incertains (que l'on croit être le bon), un paramètre fictif indiquant qu'on ignore le chiffre qu'elle contient ("0" ou ".") ou l'ensemble des chiffres qui n'enfreindraient pas les règles du jeu.

Dans l'implantation de Norvig, toutes ces possibilités sont utilisées pour représenter les cases et leur contenu à un état donné. Dans notre cas, que ce soit pour le hill-climbing, le recuit simulé ou nos heuristiques, nous n'avons utilisé que les trois premières de ces quatre possibilités de représentation du contenu d'une case.

1.2 État de départ et état but

L'état de départ est celui dont les cases initiales du sudoku ont chacune un chiffre entre 1 et 9 comme valeur alors que les autres cases (les cases vides) ont une valeur fictive ("0" ou ".") indiquant qu'elles sont vides.

L'état but est l'état dans lequel chaque case contient un chiffre unique entre 1 et 9 qui est aussi unique au sein de la ligne, la colonne et le bloc contenant cette case.

1.3 Relation de successeur

Dans l'implantation de Norvig, la relation de successeur consiste, dans une case contenant plusieurs chiffres, à en choisir un et à le retirer des chiffres possibles de toutes les cases au sein de la ligne, la colonne et le bloc contenant cette case. Dans le cas où trouver une telle case n'est plus possible, on revient en arrière afin de tester une autre possibilité.

Dans nos implémentations, la relation de successeur correspond à un échange de valeurs (un chiffre de chaque côté) entre deux cases d'un même bloc parmi les cases qui étaient vides au départ.

1.4 Coût d'étape

Le coût d'étape est de 1 pour chaque tentative de placer une chiffre dans une case (Norvig) ou de 1 pour chaque échange de valeurs entre deux cases (nos implémentations).

2 Comparaison des algorithmes

2.1 Nombre de noeuds explorés

En ce qui concerne le nombre de noeuds explorés, nous ne pouvons que constater que l'implantation de Norvig, faisant usage de la propagation de contrainte et du retour arrière, est de loin celle visitant le moins de noeud. Sur les 100 grilles de la liste fournie, l'algorithme visite en moyenne 300 noeuds avant de trouver la solution. En comparaison, le hill climbing tend à visiter autour de 2000 noeuds alors que le recuit simulé utilisant le hill climbing visite tellement de noeuds par grille qu'il est trop laborieux de tenter d'en calculer la moyenne. Notre première heuristique, un recuit simulé basé sur l'article de Lewis qui calcule la fonction de coût avec le nombre de chiffres manquants dans les unités, visite en moyenne près d'un million de noeuds. Enfin, notre seconde heuristique qui calcule le coût selon le nombre de cases en conflit, visite au maximum 7000 noeuds.

2.2 Pourcentage de réussite

Le pourcentage de réussite varie grandement d'un algorithme à l'autre. L'algorithme de Norvig fonctionne très bien, avec un taux de réussite de 100% sur la liste de 100 grilles fournie. Le hill climbing, malheureusement, termine toujours dans un minimum local et n'arrive donc à compléter aucune des grilles. Comme mentionné au paragraphe précédent, le recuit simulé utilisant hill climbing prend tellement de temps à compléter une grille qu'il fût trop laborieux d'effectuer les tests. Notre première heuristique a un taux de succès de 100%, ce qui est logique comme nous bouclons jusqu'à ce que la solution soit trouvée, alors que notre deuxième heuristique ne réussit à résoudre aucune grille car abandonne pour cause de manque de progrès sur une trop longue période avant de trouver la solution.

3 Exécution des différentes implémentations

Nous avons choisi d'inclure l'ensemble des différentes implémentations au sein du même programme, avec l'ajout d'un argument en ligne de commande afin de choisir l'implémentation à exécuter. Les différentes implémentations peuvent être exécutées de la façon suivante :

- Norvig : python sudoku.py norvig
- Hill climbing : python sudoku.py hc
- Recuit simulé : python sudoku.py sa
- Heuristique 1 : python sudoku.py sa_heur1
- Heuristique 2 : python sudoku.py sa_heur2