backtracking sur l'échiquier

Novembre 2012
dernière compilation 21 novembre 2017

Résumé

Il s'agit d'implanter un grand classique de la programmation par retours en arrière. Le placement des huit reines sur l'échiquier, le parcours du cavalier d'Euler, ou encore, la recherche du bon compte. Le sujet se termine par une question facultative d'analyse rétrograde : la construction de quelques tables de finale.

1 Préliminaire

Un algorithme de backtracking parcourt récursivement un arbre abstrait à la recherche des solutions d'un problème. Une méthodologie qui consiste à faire travailler la pile de votre ordinateur à la place de vos neuronnes...Il ne faut pas s'attendre à des miracles, les temps de calcul sont le plus souvent exponentiels et parfois pire encore. Une approche brutale qui permet d'obtenir des succès où même les génies peuvent échouer ! C'est le cas des célèbres puzzles échiquéens inventés par Leonhard Euler (1707-1783) et Carl Friederich Gauss (1777-1855). Pour cette séance, je vous propose de traiter un des trois classiques : le cavalier d'Euler, les huit Dames de Gauss et le jeu du compte est bon. Dans tous les cas, il s'agit d'écrire au moins un programme récursif en donnant des précisions sur l'aspect complexité et temps de calculs. Les plus rapides pourront traiter un sujet connexe : la construction des tables de finales par analyse rétrograde.

2 Le cavalier Eulérien

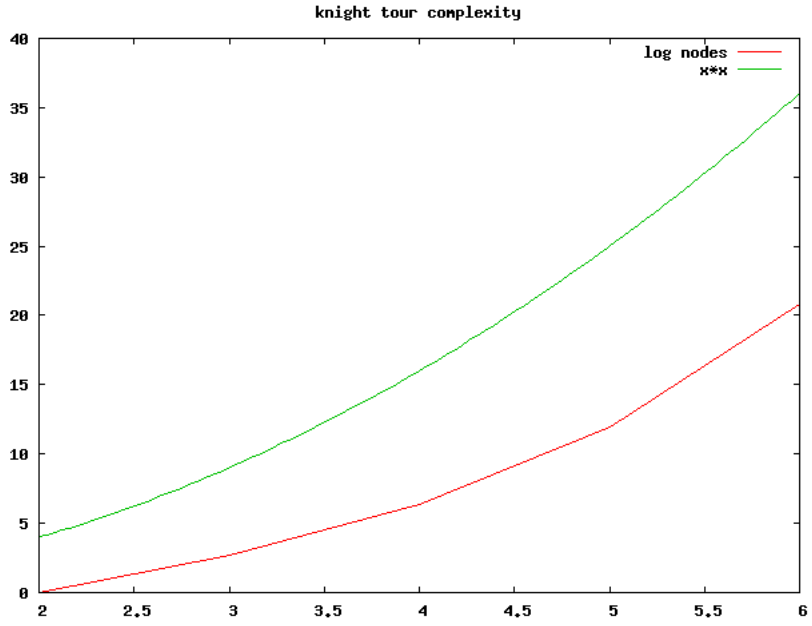
Il s'agit de compter les parcours fermés d'un cavalier sur un échiquier qui passent une et une seule fois par toutes les cases de l'échiquier avant de revenir sur la case de départ. Le problème (Knight tour problem.) se généralise de façon évidente aux échiquiers de tailles arbitraires. Dans la terminologie des graphes, il s'agit de cycle Hamiltonien.

1. Ecrire un algorithme de backtracking pour résoudre le problème d'Euler en dimension n.
2. Préciser les temps de calcul de votre implantation pour les dimensions $n = 3$, 4, 5 etc...

TABLE 1 – Dans son fameux "bréviaire des échecs" de 1933, le maître Xavier Tartakover donne une des nombreuses solutions du problème d'Euler.

14	29	34	55	12	27	24	49
35	56	13	28	33	50	11	26
30	15	54	51	58	25	48	23
41	36	57	32	61	52	63	10
16	31	40	53	64	59	22	47
37	42	1	60	19	62	9	6
2	17	44	39	4	7	46	21
43	38	3	18	45	20	5	8

3. Un échiqier de dimension impaire ne possède pas de solution. Pourquoi ?
4. Extrapoler une formule pour estimer le temps de calcul d'un échiquier 6x6.
5. Combien de temps vous faut-il pour obtenir les 9862 solutions de l'échiquer 6x6 ?
6. Comparer les performances avec ma version "bitboard".
7. Modifier votre algorithme pour qu'il examine en priorité les cases les moins accessibles.
8. Quel est l'effet de cette heuristique ?



Le nombre (correct) de solutions 13,267,364,410,532 a été calculé par Brendan McKay (1993) suite aux erreurs publiées par deux informaticiens professionnels.

3 Les huit Dames de Gauss

Comment placer huit dames sur un échiquier sans qu'aucune ne puisse être prise par une autre. En 1848, Gauss avait trouvé 72 solutions. Vous n'êtes pas Gauss mais vous avez un ordinateur !

1. Ecrire un algorithme `int Gauss( int col[], int n )` de recherche des solutions du puzzle des reines. Il y a une dame par rangée, $col[r]$ mémorise la colonne de la dame placée sur la rangée numéro r.
2. Gauss avait-il raison ?
3. Mesurer les temps de calcul en fonction de n.
4. Extrapoler une formule de temps de calcul.
5. Utiliser le site de N. Sloane pour retrouver des informations sur le sujet.
6. Comparer les performances avec ma version "bitboard".
7. Modifier la version qbit.c pour afficher les solutions.

Vous pouvez consulter l'article histo-gauss.pdf de P. Campbell pour un peu plus de rigueur sur les aspects historiques.

4 statistique sur les noeuds

Afin de prendre la bonne mesure de la puissance de calcul des machines :

1. Modifier vos codes pour retourner le nombre de noeuds parcourus.
2. Le nombre de noeuds par seconde.
3. Une statistique du nombre de noeuds parcourus par niveau.
4. Utiliser les signaux pour obtenir les statistiques à la demande en cours d'exécution.

5 Le compte est bon

Les étudiants allergiques aux 64 cases traiteront le jeu du compte est bon, la phase arithmétique du célèbre jeu télévisé des chiffres et des lettres. Rappelons qu'il s'agit d'approcher au plus prés un nombre de trois chiffres tiré au hasard entre 100 et 999 à l'aide des quatre opérations usuelles et de 6 autres nombres tirés au hasard dans un ensemble de plaques : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 25, 50, 75 et 100. Tous les calculs doivent être effectués dans l'ensemble des nombres naturels.

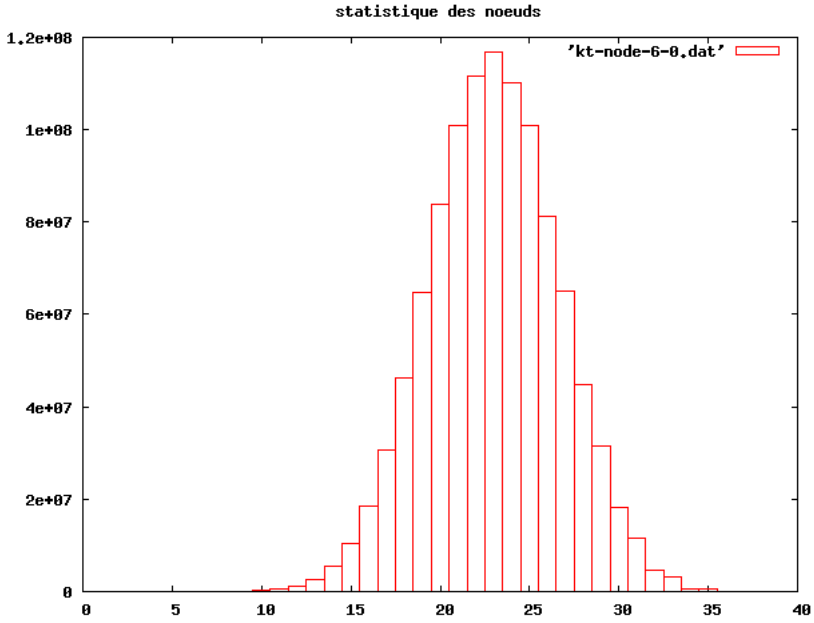


FIGURE 1 – Nombre de noeuds visités en fonction du niveau

1. Ecrire une procédure `void LeBonCompte(v, t, n)` pour trouver le bon compte v avec les nombres t[0], t[1],. . .,t[n-1].
2. Etudier le temps de calcul.
3. Faire des statistiques.

6 Table de Finale

Il ne s'agit plus de backtracking mais d'analyse rétrograde. L'analyse rétrograde des finales se fait en partant des positions dont le résultat est connu, puis de faire remonter cette analyse vers l'ensemble de toutes les positions. Du point de vue, algorithmique, on parle de programmation dynamique.

1. Décrire une représentation compacte des finales : Roi+Tour contre Roi.
2. Construire la table de finale.

7 Liens utiles

— déplacement des pièces aux échecs
— Table de Finale
— Les tables de Nalimov en ligne
— knight tour

— ma version bitboard
— on-line encyclopedia of integer sequence
— Gauss et les 8 dames