Hitori avec recuit simulé

Introduction

Ce projet est une implémentation de l'algorithme du recuit simulé afin de résoudre une grille du jeu hitori. Le but est de transformer une grille d'origine en noircissant certaines cases afin de respecter les 3 règles suivantes :

- Chaque ligne et chaque colonne ne doit contenir qu'une seule occurrence d'un chiffre donné.
- Les cases marquées ne doivent pas être adjacentes horizontalement ou verticalement, elles peuvent être en diagonale.
- Les cases non marquées doivent toutes être connectées entre elles par adjacence horizontalement ou verticalement. Nous avions à résoudre cette grille en utilisant une des méta-heuristiques suivantes : Recuit simulé ; Algorithmes Génétiques ; Optimisation par Essaim de Particules. J'ai choisi le recuit simulé car c'est selon moi la méta-heuristique la plus adapté au problème

Solution

Afin de résoudre n'importe quelle grille de Hitori, j'ai décidé d'initialiser un tableau contenant les coordonnées des éléments qui possèdes un élément dupliqué sur sa ligne ou sa colonne. Cette liste nous permettras lorsque nous allons générer un nouvel état de ne pas modifier une case dont le changement n'impacteras pas la solution finale. Une fois les éléments communs rassemblés, on peut commencer à résoudre la grille. On effectue la résolution tant que notre "score" est supérieur à 0 et on effectue les étapes suivantes :

- On recopie notre grille
- On modifie une case aléatoire et on calcul son "score"
- si on obtient un meilleur score ou si un nombre aléatoire est inférieur à l'inverse de 1 + l'exponentielle du delta du score / T
    - On vérifie la validité de la grille (toutes les cases blanches sont connectées) et si elle l'est on sauvegarde le score et le nouvel état de la grille. Si le score vaut 0 alors on arrête le programme.
- On fait baisser la température en multipliant notre température par un facteur préalablement renseigné
- Si notre température atteint un certain seuil (donné en paramètre) alors on augmente la température à une valeur renseignée en paramètre.

Utilisation

```
java -jar Hitori.jar filename size nbALancer startTemperature decay [iterPerTemp] [seuilTemp] [resetTemp] [showGrid=1|0]
```

Exemple : `java -jar Hitori.jar 15.txt 15 10 200 0.9995 250 0.000001 20 1`

Résultats

Configuration moyenne, on va s'en servir comme grille de référence

- T0: 200
- D: 0.9995
- I: 250
- ST: 0.000001
- TR: 20

	5x5	8x8	10x10	12x12	15x15	20x20
1	41ms	37ms	260ms	292ms	93 404ms	44 726ms
2	2ms	76ms	118ms	222ms	24 354ms	139 961ms
3	1ms	31ms	103ms	119ms	58 690ms	104 358ms
4	5ms	58ms	251ms	128ms	24 585ms	44 516ms
5	9ms	9ms	324ms	112ms	15 130ms	92 281ms
6	3ms	41ms	119ms	98ms	76 174ms	21 828ms
7	6ms	56ms	14ms	103ms	50 773ms	127 000ms
8	8ms	39ms	77ms	149ms	50 099ms	20 697ms
9	7ms	163ms	51ms	212ms	15 100ms	21 095ms
10	6ms	7ms	31ms	206ms	6 469ms	59 951ms
MOY	**9ms**	**52ms**	**135ms**	**164ms**	**41 477ms**	**67 641ms**

On fait baisser la température vite afin de souvent la remettre à 0

- T0: 100
- D: 0.989
- I: 250
- ST: 0.000001
- TR: 50

On note une instabilité dans les temps de réponse ainsi qu'une moyenne qui augmente rapidement en fonction de la difficulté de la grille

	5x5	**8x8**	**10x10**	**12x12**	**15x15**	**20x20**
1	24ms	46ms	684ms	155ms	64 787ms	572 444ms
2	1ms	92ms	89ms	390ms	69 660ms	518 355ms
3	23ms	88ms	132ms	368ms	14 889ms	412 874ms
4	6ms	66ms	38ms	447ms	98 569ms	xxxx
5	1ms	77ms	173ms	77ms	49 653ms	xxxx
6	13ms	104ms	143ms	499ms	40 404ms	xxxx
7	16ms	47ms	356ms	245ms	53 980ms	xxxx
8	2ms	25ms	54ms	239ms	90 343ms	xxxx
9	5ms	95ms	138ms	169ms	238 520ms	xxxx
10	1ms	113ms	634ms	56ms	13 671ms	xxxx
MOY	**9ms**	**75ms**	**244ms**	**264ms**	**73 447ms**	**501 224ms**

On ne remet à 0 la température que rarement

- T0: 300
- D: 0.9999
- I: 400
- ST: 0.000001
- TR: 50

Les résultats sont très stables mais sont cependant plus lent que notre premier test.

	5x5	8x8	10x10	12x12	15x15	20x20
1	45ms	45ms	257ms	148ms	50 765ms	70 527ms
2	6ms	29ms	182ms	96ms	51 560ms	72 410ms
3	6ms	40ms	385ms	183ms	50 477ms	74 187ms
4	11ms	114ms	31ms	86ms	51 205ms	72 218ms
5	3ms	86ms	349ms	399ms	50 326ms	72 142ms
6	4ms	90ms	168ms	227ms	50 773ms	73 219ms
7	3ms	66ms	349ms	52ms	50 382ms	71 645ms
8	1ms	28ms	81ms	252ms	50 181ms	73 257ms
9	4ms	50ms	188ms	157ms	50 559ms	73 342ms
10	6ms	15ms	130ms	337ms	52 027ms	71 449ms
MOY	**9ms**	**56ms**	**212ms**	**194ms**	**50 825ms**	**72 439ms**

On utilise que des températures basses

- T0: 10
- D: 0.9995
- I: 250
- ST: 0.000001
- TR: 5

On peut voir que les résultats pour les grilles jusqu'à la 15x15 sont nettement inférieur, cependant pour la grille 20x20 le temps à beaucoup augmenté. On peut déduire qu'il faut un minimum de température maximal afin de pouvoir résoudre dans un temps correct les grandes grilles et que pour les plus petites grilles, avoir une température élevée est inutile.

	5x5	**8x8**	**10x10**	**12x12**	**15x15**	**20x20**
1	51ms	25ms	294ms	112ms	36 026ms	145 102ms
2	4ms	92ms	64ms	194ms	20 495ms	49 287ms
3	8ms	25ms	103ms	85ms	82 990ms	38 517ms
4	12ms	4ms	140ms	192ms	5 050ms	145 101ms
5	6ms	23ms	327ms	96ms	36 099ms	177 592ms
6	2ms	57ms	74ms	356ms	43 850ms	60 251ms
7	2ms	8ms	205ms	56ms	12 383ms	113 317ms
8	13ms	169ms	84ms	89ms	4 559ms	144 252ms
9	5ms	26ms	89ms	352ms	20 597ms	28 033ms
10	2ms	88ms	42ms	44ms	28 000ms	112 178ms
MOY	**10ms**	**52ms**	**142ms**	**158ms**	**29 005ms**	**101 363ms**