Rapport modélisation programmation par contraintes

Pierre Kouyoumdjian

May 31, 2022

1 Introduction

Pour ce projet de modélisation programmation par contraintes le but était de résoudre le problème all-interval-series

2 Solution et comparaison

Pour lancer les jars, il faut rentrer la commande

```
java -jar fichier.jar n
```

n représentant le nombre de variable

2.1 Sans table

Sans table pour 5 variables nous avons ces résultats :

```
N = 5
AC_REGIN { backtrack = 113, node = 64, fails = 49, solutions = 8, time = 0.032554206 }
AC { backtrack = 113, node = 64, fails = 49, solutions = 8, time = 0.011525681 }
AC_ZHANG { backtrack = 113, node = 64, fails = 49, solutions = 8, time = 0.007407118 }
BC { backtrack = 115, node = 65, fails = 50, solutions = 8, time = 0.007266907 }
FC { backtrack = 137, node = 76, fails = 61, solutions = 8, time = 0.003969172 }
NEQS { backtrack = 135, node = 75, fails = 60, solutions = 8, time = 0.004294687 }

N = 10
AC_REGIN { backtrack = 86805, node = 43698, fails = 43107,
solutions = 296, time = 1.7111145 }
AC { backtrack = 86805, node = 43698, fails = 43107,
solutions = 296, time = 1.1637957 }
AC_ZHANG { backtrack = 86805, node = 43698, fails = 43107,
solutions = 296, time = 1.1042154 }
BC { backtrack = 101889, node = 51240, fails = 50649,
solutions = 296, time = 1.1629735 }
FC { backtrack = 228659, node = 114625, fails = 114034,
solutions = 296, time = 1.318281 }
NEQS { backtrack = 267481, node = 134036, fails = 133445,
solutions = 296, time = 2.1592014 }

N = 12
AC_REGIN { backtrack = 2247481, node = 1125068, fails = 1122413,
solutions = 1328, time = 41.84171 }
AC { backtrack = 2247481, node = 1125068, fails = 1122413,
solutions = 1328, time = 38.97535 }
```

```
AC_ZHANG { backtrack = 2247481, node = 1125068, fails = 1122413,
solutions = 1328, time = 38.76494 }
BC { backtrack = 2567239, node = 1284947, fails = 1282292,
solutions = 1328, time = 34.787975 }
FC { backtrack = 8027563, node = 4015109, fails = 4012454,
solutions = 1328, time = 63.092735 }
NEQS { backtrack = 8861893, node = 4432274, fails = 4429619,
solutions = 1328, time = 95.87481 }
```

2.2 Avec table

Avec table pour n = 5 :

```
N = 5
AC_REGIN { backtrack = 55, node = 35, fails = 20,
solutions = 8, time = 0.022174615 }
AC { backtrack = 55, node = 35, fails = 20,
solutions = 8, time = 0.006561645 }
AC_ZHANG { backtrack = 55, node = 35, fails = 20,
solutions = 8, time = 0.006400282 }
BC { backtrack = 67, node = 41, fails = 26,
solutions = 8, time = 0.006418365 }
FC { backtrack = 83, node = 49, fails = 34,
solutions = 8, time = 0.003934875 }
NEQS { backtrack = 101, node = 58, fails = 43,
solutions = 8, time = 0.005360896 }

N = 10
AC_REGIN { backtrack = 19719, node = 10155, fails = 9564,
solutions = 296, time = 0.79611206 }
AC { backtrack = 19719, node = 10155, fails = 9564,
solutions = 296, time = 0.5711924 }
AC_ZHANG { backtrack = 19719, node = 10155, fails = 9564,
solutions = 296, time = 0.36010817 }
BC { backtrack = 21307, node = 10949, fails = 10358,
solutions = 296, time = 0.4094257 }
FC { backtrack = 145815, node = 73203, fails = 72612,
solutions = 296, time = 1.3291813 }
NEQS { backtrack = 193885, node = 97238, fails = 96647,
solutions = 296, time = 1.9218069 }

N = 12
AC_REGIN { backtrack = 391875, node = 197265, fails = 194610,
solutions = 1328, time = 10.613299 }
AC { backtrack = 391875, node = 197265, fails = 194610,
solutions = 1328, time = 9.968451 }
AC_ZHANG { backtrack = 391875, node = 197265, fails = 194610,
solutions = 1328, time = 9.964307 }
BC { backtrack = 537339, node = 269997, fails = 267342,
solutions = 1328, time = 11.162833 }
FC { backtrack = 5887369, node = 2945012, fails = 2942357,
solutions = 1328, time = 68.07839 }
NEQS { backtrack = 7124349, node = 3563502, fails = 3560847,
solutions = 1328, time = 100.00178 }
github
```

2.3 Comparaison

On remarque que plus on augmente n plus le solver prend du temps. On peut remarquer que sans table on passe de 0.003969172s pour le meilleur de n = 5 à 34.787975s pour n = 12.

Pour celui qui utilise la table, n = 5 donne les résultats en 0.003934875s et pour n = 12 nous avons les résultats en 9.964307s.

On voit alors que la différence sans table/table est d'un facteur 3.5. Ce qui est vraiment conséquent. On peut donc dire qu'il est intéressant d'utiliser les tables à partir de n = 12. A n = 10 la différence reste minime par rapport à n = 12