**CALVI Alexis**

**GARIN Lucas**

Optimisation discète

PROJET METAHEURISTIQUES

Problème des dames

Sommaire

1. Introduction
2. Architecture
3. Fitness
4. Recherche des voisins
5. Recuit simulé
6. Tabou
7. Génétique
8. Introduction

Le problème des dames est un problème d’optimisation discrète, on doit rechercher une solution optimale (ici il y en a plusieurs) dans un très grand ensemble de solution. Le nombre de solutions peut être très grand et nous devons donc trouver des algorithmes qui nous permettent de parcourir l’ensemble des solutions intelligemment afin de ne pas toutes les parcourir.

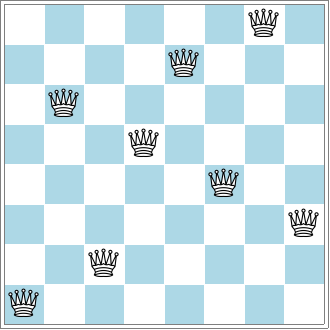
Le problème : Placer n dames sur un plateau de n cases de côté sans aucune ne puissent se manger entre elles, c’est-à-dire : aucune sur la même ligne, colonne ou diagonale.

Afin de répondre à ce problème nous avons utilisé trois différents algorithmes métaheuristiques : recuit simulé, tabou et un algorithme génétique.

1. Architecture

Nous avons décidé de modéliser les solutions en 1 seule dimension. En effet, vu que aucune dame ne peut être sur la même ligne ou sur la même colonne, un nous suffit de trouver la bonne série de ligne correspondant à la position de la dame de chacune de colonnes. Il ne nous reste donc plus qu’a testé les diagonales.

Exemple :



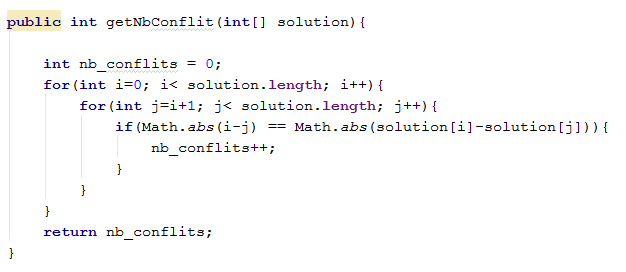
*La solution ici s’écrit : 8,3,7,4,2,5,1,6 (qui correspond à ligne de chaque dame positionnée sur les colonnes allant de gauche à droite)*

Au niveau architecture de développement, nous avons décidé de centralisé les méthodes communes aux algorithmes dans une classe abstraite afin de factorisé le code (getVoisins, firstPlateau et getNbConflits).

Les deux méthodes qui calculent le fitness et la liste des voisins d’une solutions sont essentielles dans un algorithme métaheuristique…

Fitness

Cette méthode va être appelée à chaque solution traitée par notre algorithme donc elle se devait d’être la plus optimale possible.



Nous testons les diagonales, cela nous permet également de gagner en efficacité car la complexité est améliorée.

Recherche des voisins

Nous avons choisi la transformation locale comme recherche des voisins. Il y a donc n ! voisins à chaque solution.