IT-Universitet i København

Intelligent systems programming

Project 2

Tróndur Høgnason (thgn@itu.dk)
Kristian Mohr (kvin@itu.dk)

April 15, 2017

1   Introduction

The n-queens problem is

2   QueensLogic

The two classes `QueensGUI` and `ShowBoard` are only relevant to drawing the solution and we will therefore only describe the methods in the class `QueensLogic`. `QueensLogic` is the class responsible for all logic relevant to solve the n-queens problem and has the following methods:

InitializeGame Sets N, the width and height of the game board and create a 2D int array to represent the chess board. Lastly it calls *createBDD* and *setInvalids*.

getGameBoard Return the 2D int array representing the chessboard.

createBDD Uses the javabdd package to initialize a BDD with 2.000.000 nodes and 200.000 cache as suggested. It then creates a BDD with N*N variables – one for each chessboard position. Lastly it calls `createRules`.

createRules Creates the first of the two rules for the BDD to solve the n-queens problem. The first rule is that there must be a queen in any column. It calls `noCaptureRule` to create the second rule.

noCaptureRule Creates the second rule, which specifies that no queen must be able to capture another.

position Converts a board position of the form *column, row* to variable number in the BDD between 0-63.

isInvalid Checks if setting a variable in the BDD to true makes the BDD false.

setInvalids Goes through all chessboard positions and calls isInvalid to check if placing a queen on a particular tile makes the BDD false.

setRemainingVailds Set all positions still valid on the chessboard to 1 which makes the GUI place a queen there.

insertQueen Returns without doing anything if the users is trying to place a queen in an invalid position or in a position already containing a queen. If that is not the case, it places a queen on the given position, updates the BDD and the invalid positions by calling *setInvalids*. Lastly, if there is only one remaining solution to the BDD, it calls `setRemainingValids`.

3   Rule implementation

We will go through how the two rules, at-least-one-queen-per-row and no-queen-can-capture-another are implemented. The first rule one-queen-per-row is implemented by:

1. Creating a sub BDD that is false for every column.

2. Going through every position on the chessboard and checking whether or not it is possible to place a queen there.

The second rule no-queen-can-capture-another is implemented by going through all positions in the chessboard and:

1. Create a sub BDD representing the that no queen can capture another. The BDD is initialized to true.

2. Go through all column positions with the given row position and checking that there is no queen there.

3. Go through all row positions with the given column position and checking that there is no queen there.

4. Go through all the diagonals positions with the given column and row position and checking that there is no queen there.

4   Conclusion