BLG458E
Homework-2

05.01.2024

Res. Asst. Yusuf Huseyin Sahin
sahinyu@itu.edu.tr

*C*hess is mental torture.

Garry Kasparov

- You should write all your code in Haskell language.
- Cheating is highly discouraged. If you are planning to use different libraries or functions, please ask me about it.
- In your report please describe how to run your functions.

1 Introduction: ChessBoard, Problem & Solution

This project aims to place chess pieces of five teams on a chess board of different sizes. The problem can be counted as a multi-objective approach on the N-Queens problem. In the N-Queens problem, the objective is to place N queens on an NxN chessboard so that none of the queens can attack each other. In N-Queens problem, all the pieces can be though as belonging to different teams. Our changes over the problem are; allowing the pieces to belong to same team and including also the other type of chess pieces. The only change for the other pieces is about pawn's attacking areas. In our problem, pawn is assumed to treat all pieces at its four corners.

Piece	**Value**
Bishop	3.25
Knight	3.25
Rook	5
Queen	10
Pawn	1

Table 1: Values of different pieces

In the problem there are two objectives. First, the count of total attacking positions should be minimized. Second, the value of the team with the minimum value should be maximized. For the second objective, the values of different pieces which are also used for traditional algorithms in chess programming are employed. The values of different pieces can be seen in Table 1.

An example 8x8 solution with a total of 18 attacks and a minimum value of 10 given in Figure 1.

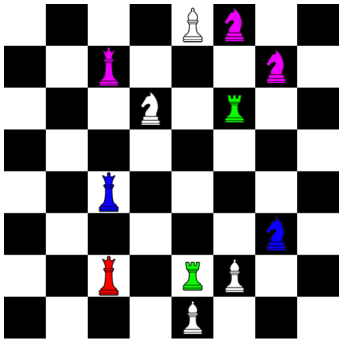

Figure 1: An example 8x8 board

2 (20 pts): Part 1 - Module Design

Create a module to define a chessboard with a given size. To do this you may create the following enumerated types:

- **Piece:** Bishop, Knight, Rook, Queen, Pawn
- **Team:** Red, Green, Blue, Purple, White
- **Chessboard:** A 2D list of tuples (Piece, Team)

3 (20 pts): Part 2 - Score and Change Functions

After defining the types, the following functions should be defined.

- **attackcount**: Total number of attacks of the given chessboard.
- **boardvalue**: Total value of the board.
- **horizontalcrossover**: For a horizontal line, change upper and lower parts of the board.
- **verticalcrossover**: For a vertical line, change left and right parts of the board.

- **deletepiece**: Delete a piece from the board.

An example of vertical crossover for the sixth tile of a 8x8 is given in Figure 2.

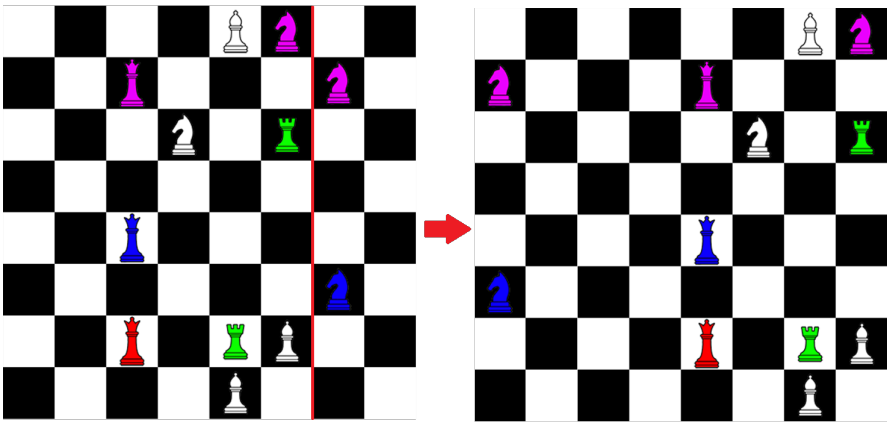

Figure 2: An example vertical crossover

4 (60 pts): Part 3 - Chessboard Tree

Create a tree containing a chessboard given in Figure 1 as the head node. For every node, 14 branches could be created using horizontalcrossover and verticalcrossover functions and 12 branches could be created using deletepiece function. For every level of tree (1-5), find the best chessboard for the both tasks.