COMS3008: Parallel Computing Assignment

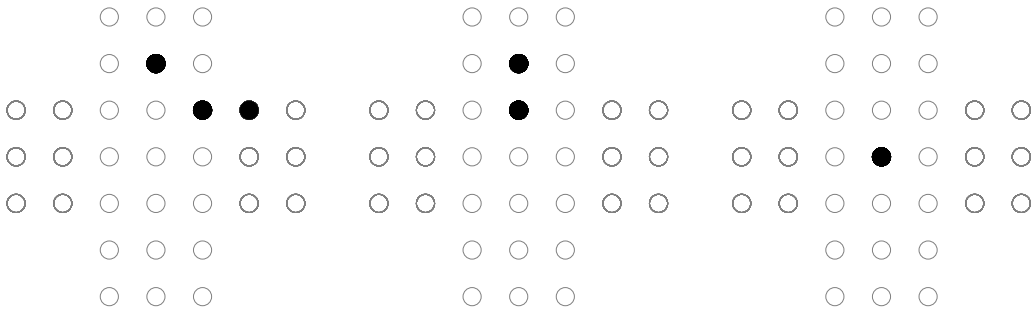
Tau Merand 908096 Vincent Varkevisser 705668

October 19, 2017

Introduction

The game of peg solitaire is a one player game played on a 33 holed cross shaped board that involves jumping pegs over other pegs, in a manner similiar to checkers. The rules are as follows:

1. A move consists of jumping a peg over an orthoganal neighbour into an empty space. The peg that was jumped over is then removed from the board.
2. Pegs can only jump onto an empty space.
3. The game is won if the final peg is in the centre space.
4. If no pegs can legally move or the final peg is not in the centre the game is lost.



(a) A start state

(b) The right most peg jumps over its left neihgbour

(c) The winning state

Figure 1: A winning set of valid moves

Backtracking Search

Recursive backtracking using depth first search was chosen as the method for state space exploration. The standard depth first algorithm is as follows:

input : An initial, valid board state
output: A sequence of moves to get from the initial state to the winning state

result ← Empty list to store moves from the initial state to the winning state *legalMoves* ← A list of all the legal moves for the current board;
state ← The current state;
foreach *m in legalMoves* **do**
    state ← The state achieved by performing m;
    if *state is winning state* **then**
        return *move*;
    else if *state has no legal moves* **then**
        return *Null*;
    else
        return BacktrackingDFS(state);
    end
end

Algorithm 1: A standard recursive backtracking using DFS

But because pegs are identical, game states where pegs are in the same position are identical, regardless of the moves taken to arrive at that state. Thus exploring the state space looking for a sequence of states leading to the winning state will probably involve evaluating the same states many times as illustrated below.

Serial Implementation

Parallel Implementation

Results