Introduction

The proposal for this project is to study abstract solitaire games with a general search engine. The game definition format will be described. Then the game solution engine will be described. Finally, a proposed list of games to be investigated will be listed. The motivation for this project is the game(s) of peg solitaire but it is hoped that the approach described here can be used to study a wide variety of solitaire games.

Game definition

A **game** is defined using the following data structure:

- A **board** is a list of integers, B. $B[i]$ is the description of position i.
- The game proceeds by a sequence of boards: $B_0 \to B_1 \to \cdots$ until no further moves are possible
- A **move** is a transformation of a board to a new board. The moves of the game are defined as a set of **transformations**.
- A **transformation** has the form $\left[[P_0, C_0], [P_1, C_1]\right]$ where P_0 $(resp. P_1)$ are lists of board positions and C_0 $(resp. C_1)$ are lists of descriptions of the board positions.
- A move can be **triggered** on a board by matching positions in P_0 with the description in C_0. If a match is made, then the new board is formed from the old board by replacing the description of positions P_1 with C_1.
- A set of **terminal descriptions**. Each terminal description is of the form $[P_0, C_0]$ as described above.

Game example: peg solitaire

For the game(s) of peg solitaire, the board is a list where each position has:

- 1, indicating that the position is filled
- 0, indicating that the position is empty
- -1, indicating that the position is "forbidden". [This allows boards to be described compactly even though the boundary of the board may be irregular.]

The moves would be of the form $\left[\left[P_0, [1,1,0]\right], \left[P_0, [0,0,1]\right]\right]$ where P_0 would range over all positions that are in a "row" or "column". Note that the abstract definition of the game does not define rows or columns. These would be implicit in the set of possible moves.
The terminal positions would be of the form $[B, C]$ where B is the entire board and C is a list generally with all $0's$ and a single 1 in a "winning" position (with forbidden positions preserved).

Search engine

The design of the engine is part of the project investigation. The first attempt is to use a depth-first search algorithm with backtracking. If this is successful, other algorithms can be investigated. Breadth-first search to find the shortest path is unlikely to be successful due to combinatorial explosion, however, other search methods with a probabilistic aspect could be explored.

Games to be investigated

The motivation for this project are peg solitaire games. These games could be played on boards such as the following:

- Rectangular boards (hopefully this is the easiest case)
- Rectangular boards with forbidden positions. This is the standard for the commercial games played.
- Triangular or hexagonal boards.
- Topologically interesting boards: cylinder, Möbius strip, torus, Klein bottle
- Higher dimensional boards

Other games could be investigated such as the "15-game" where the board is a 4 x 4 matrix with tiles labeled $1..15$ and one empty position. Each tile adjacent to the empty position can be moved to that position. The goal of the game is to arrive at a particular configuration. Note that a depth-first search will preclude cycles but may not find the most efficient solution.