Artificial intelligence assignment 1; practicing basic and informed search algorithms.

Game of Chess

Problem

Consider the game of Chess (https://en.wikipedia.org/wiki/Chess). Approximately, how many different states can a chess game have? You may assume

1. We do not consider pawn promotion
2. We consider each pawn is a distinct piece
3. We consider only states with no captured pieces, i.e. all states will have 32 pieces
4. We do not consider pawn promotion, castling, or any other special cases.

State any other assumptions made and show your work. Your answer does not have to be mathematically correct, but should be in the right magnitude for the assumptions you have made and make sense based on your math.

Answer

Additional assumtions will be:

1. Pawns can not exist in the 1^{st} or 8^{th} rank
2. Pawns can not stack (i.e. two of the same colored pawns cannot be on the same column)
3. Black/White have one bishop that resides in the black tiles and one bishop that resides on the white tiles
4. How the position came to be is not relevent
5. Kings can be within one tile of eachother

We will start by calculating the number of possible positions of the pieces which do not have tile restrictions. Then, we will place the pieces that do have possible restriction; that is, the bishops and pawns.
First we start with a single rook. Without any other pieces on the board, it has 64 possible positions it can exists on.
Then, we go to a second rook. Since one of the tiles is occupied by the first placed rook, the second rook only has 63 possible tiles ot exist on.
Similarilly, the third rook will only have 62 tiles that it can possible exist on.
At this stage, we can recognize a pattern: As tiles get occupied by previous pieces, the number of available tiles reduces by one.
The number of states the pieces (minus the pawns and bishops) can be represented by the following product series.

$$s_{tot1} = \prod_{i=0}^{p-1} (64 - i) \tag{1}$$

Where s_{tot1} are the total number of possible states for the pieces without tiles restrictions and p is the number of pieces placed.
There are a total of 12 chess pieces without the pawns and bishops. Therefore,

$$s_{tot1} = \prod_{i=0}^{11}(64 - i) = 64 * 63 * ... * 53 \approx 1.573 * 10^{21} \tag{2}$$

Now, we must place the pawns and bishops.
The pawns are restricted from the 2^{nd} to the 7^{th} rank and to one of the columns.
This gives each of the pawns 6 tiles to exist on. That means, without any obstructions, pawns can have a total of

$$s_{tot2} = 6^8 + 5^8 \approx 2.07 * 10^6 \tag{3}$$

where s_{tot2} is the total number of states for the pawns. Note: we will not filter out states where other pieces are occupying all possible tiles in a pawn's column since this is negligible considering the magnitude of number in Equation 2.
The bishops are each restricted to either the white or the black tiles. This gives each of the bishops 32 tiles to exist on. That means, without any other pieces taking any tiles from the bishops, we can have a total of

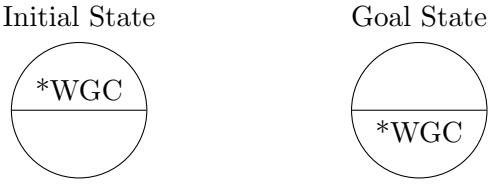
$$s_{tot3} = 32^2 + 31^2 \approx 9.84 * 10^5 \tag{4}$$

The results from Equations 3 and 4 are very negligible compared to the result from Equation 2. We can then say there will be approxamilly $1.573 * 10^{21}$ possible states of the chess position given the initial restrictions.

Wolf, Goat, and Cabbage Problem

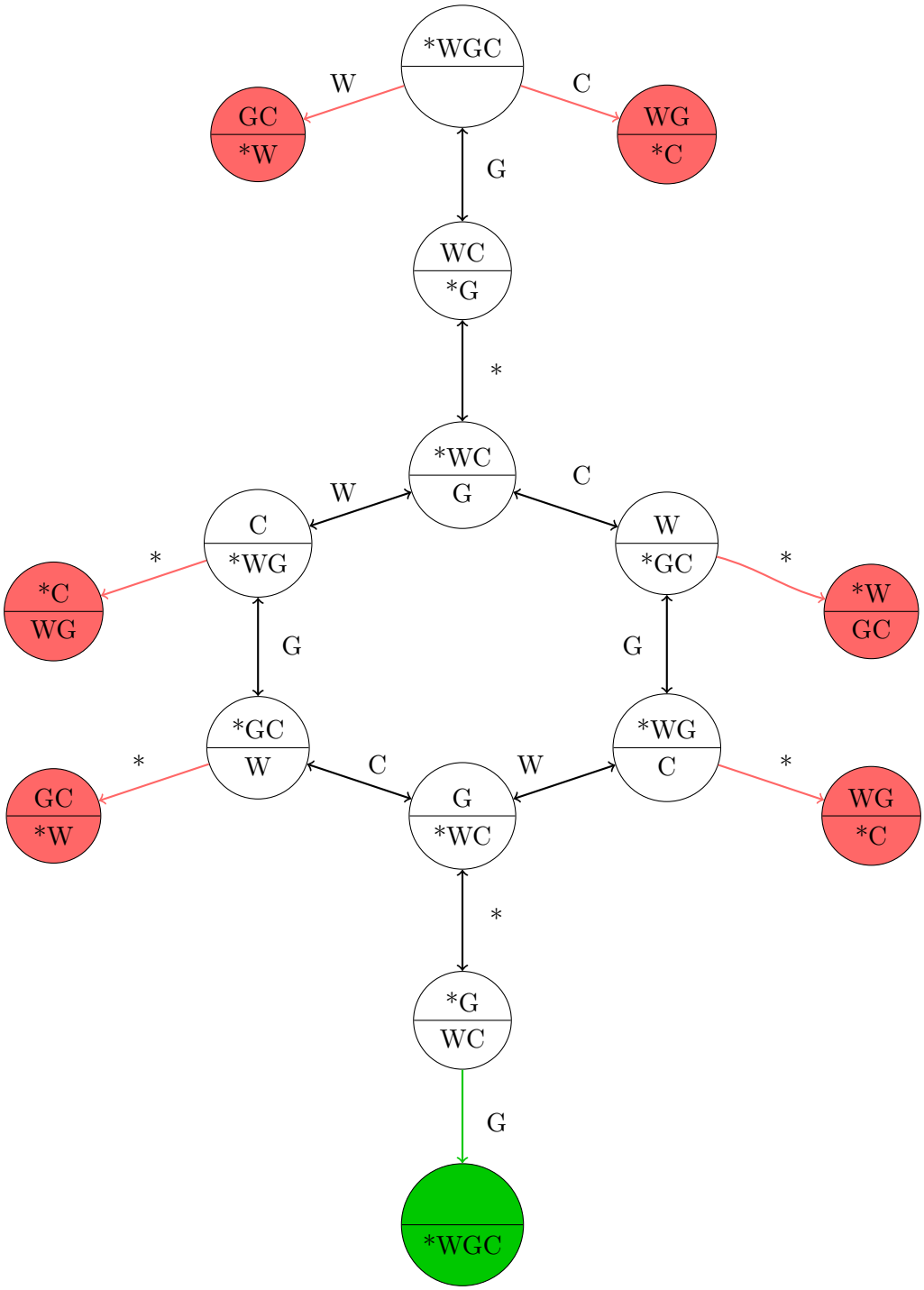
Problem

A farmer has just bought a wolf, a goat, and a cabbage. They must cross the river to get home, but their small rowboat can only carry the farmer and 1 animal or vegetable at a time. However, the farmer cannot leave the goat unsupervised with the cabbage, or the goat will eat the cabbage. Similarly, the farmer cannot leave the goat unsupervised with the wolf, or the wolf will eat the goat. Draw the full search space starting from the initial state below. Indicate with different colors or labels which states would not be further explored because something gets eaten, as well as which states would not be further explored because they are repeats of previous states (to avoid cycles)



Answer

Here is the full search space where $*$ represents the farmer, W represents the wolf, G represents the goat, C represents the cabbage. In the transitions, the letter shown next to the edge is the animal or vegetable crossing the river; the $*$ alone is the farmer crossing the river on his own.



Search Algorithms on a State Space Graph

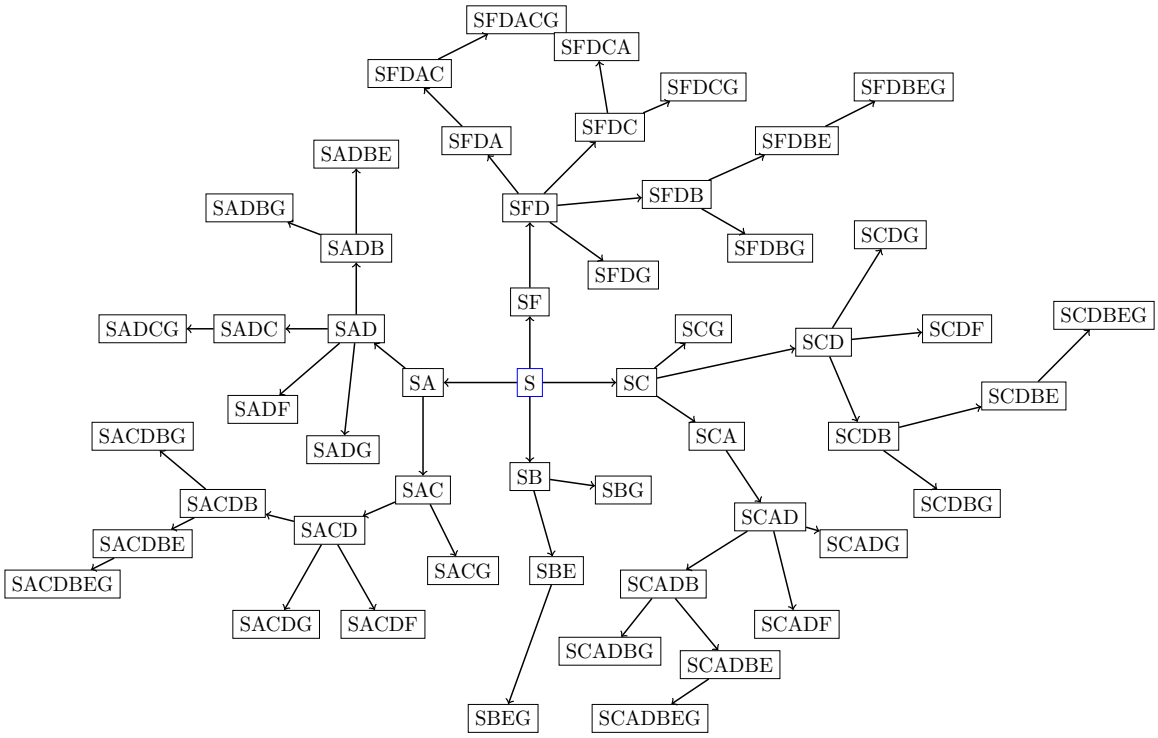
Problem

Consider the following state space graph with Initial State S and Goal State G

1. Draw out the complete search tree, ignoring cycles by not allowing the same node in the path more than once. The tree has been started below to show formatting.
2. List the Node exploration order for a Breadth First Search. For grading simplicity, when there is a tie, explore the nodes alphabetically.
3. List the Node exploration order for a Depth First Search. For grading simplicity, when there is a tie, explore the nodes alphabetically.
4. The table below shows the estimated distance from each node to the goal. Using these as the heuristic h(n), list the Node exploration order for a Best-First Search. For grading simplicity, when there is a tie, explore the
5. Using the same table as the heuristic h(n), list the A* Node exploration order. For grading simplicity, when there is a tie, explore the nodes alphabetically. nodes alphabetically.

Answer

1. Complete search tree without repeating nodes. Tree starting at S.



2. Node exploration order for Breadth First Search.

S	SA	SB	SC	SF	SAC	SAD	SBE	SBG

3. Node exploration order for Depth First Search.

S	SA	SAC	SACD	SACDB	SACDBE	SACDBEG

4. Node exploration order for Best-First Search.

S	SC	SCG

5. Node exploration order for A* Search.

S	SC	SCG

Search Trees on a Grid

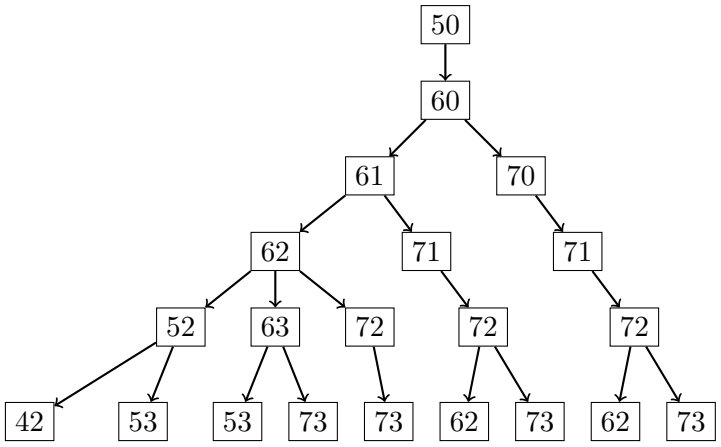
Poblem

Consider the grid below with start state 50 and goal state 49:
Draw the search tree for the following algorithms a-d assuming:

1. Black squares are walls that cannot be passed through.
2. Break ties in the order N-E-S-W.
3. Each step has a uniform cost.
4. The heuristic is the Manhattan distance to the goal. (Don't count black wall tiles, but for heuristic estimate, go through them)

Answer

1. Breadth-First Search (5 layers). Nodes are search from left to right on each layer.



2. Depth-First Search. For this problem I will list the nodes visited on a table from left to right since a tree will be ridiculously long.

50	60	61	62	52	42	32	33	23	13	3	4	5	6
16	26	36	37	38	28	18	8	9	19	29	39	49	

3. Beam Search with W=2 (5 layers)
4. A*
5. What is the optimal path found by A*