1 MCTS

1.1 MCTS Review

Upper-confidence bound The UCB formula is:

$$\text{UCB}(N, N_k, U_k) = \begin{cases} \frac{U_k}{N_k} + C\sqrt{\frac{\log N}{N_k}} & \text{if } N_k > 0 \\ \infty & \text{otherwise} \end{cases}$$

Here, N is the visitation count to the parent node; Nk is the visitation count to a child node; Uk is the total utility accumulated at the child.

MCTS with UCT

1. **Selection** Starting at the root of the search tree, choose moves down to a leaf node according to UCT. Return the leaf node.
2. **Expansion** Grow the search tree from the leaf node by generating all its children. If the leaf node is already at the end of the horizon, skip this step.
3. **Simulation** Perform Monte-Carlo playout from the leaf node. Note that no new node is added to the search tree.
4. **Back-propagation** Use simulation's result to update all visitation counts and cumulative rewards, going up to the root.

1.2 MCTS Practice

Consider the reward maximization problem in the grid on the right.
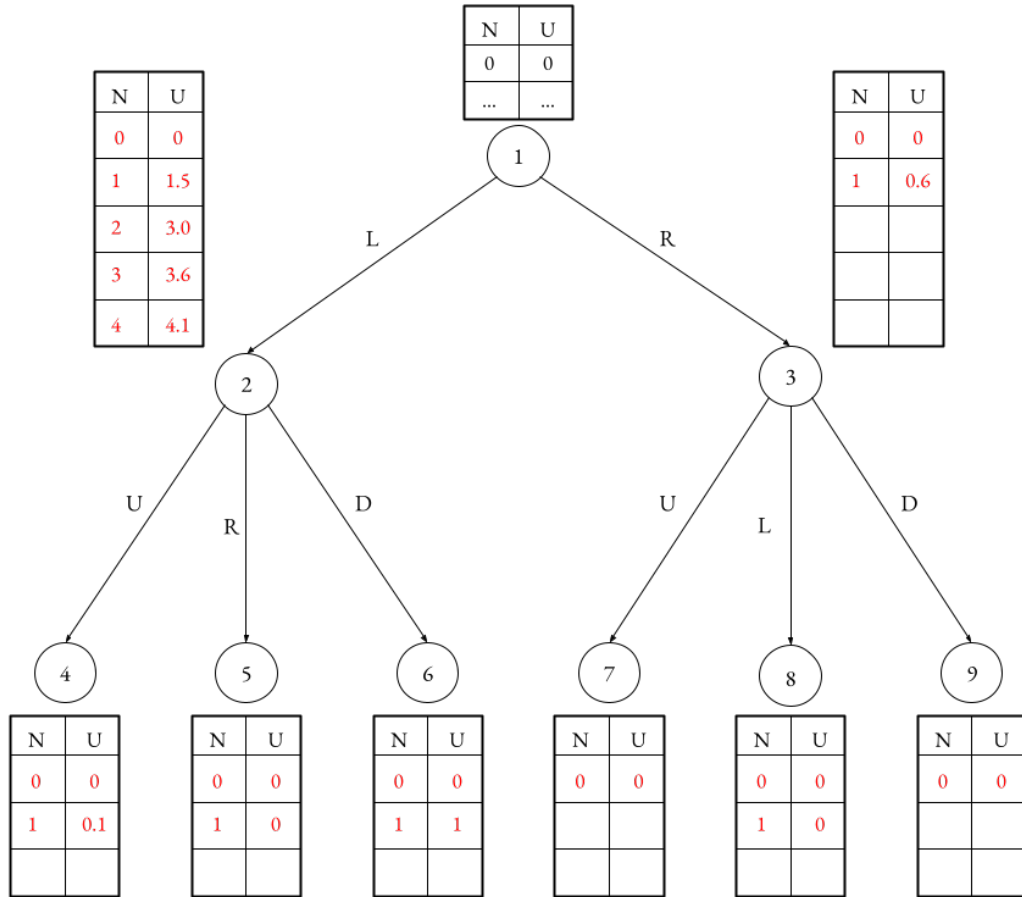
- The agent starts at s_0 at the center
- The agent can move left, right, up, or down, except that it cannot move onto black grid cells
- Entering each grid cell yields a reward (specified within the box)
- Assume a horizon of 2
- Assume that $C = 1.0$

Simulate running MCTS and fill in the search tree below until you run out of cells (note: some table cells may remain empty). Assume that the random choices are $[0, 2, 1, 2]$ (select the first child $\rightarrow$ the third child $\rightarrow$ the second child $\rightarrow$ the third child). Break ties during the selection phase by selecting the leftmost child in the tree.

The following computations might be helpful:

- $\sqrt{\frac{\log(n)}{1}} - \sqrt{\frac{\log(n)}{n-1}} \leq 0.5$ for $2 \leq n \leq 4$
- $\sqrt{\frac{\log(n)}{2}} - \sqrt{\frac{\log(n)}{n-2}} \leq 0.5$ for $3 \leq n \leq 8$

N	U
0	0
...	...

(1)

N	U
0	0
1	1.5
2	3.0
3	3.6
4	4.1

N	U
0	0
1	0.6

L

R

(2)

(3)

U

R

D

U

L

D

(4)

(5)

(6)

(7)

(8)

(9)

N	U
0	0
1	0.1

N	U
0	0
1	0

N	U
0	0
1	1

N	U
0	0

N	U
0	0
1	0

N	U
0	0

# 2   CSP

2.1   CSP Review

1. **Constraint Graph** Graph where nodes correspond to variables of a problem and edges connect any two variables participating in a constraint.
2. **Arc Consistency** A variable in a CSP is arc-consistent if every value in its domain satisfies the variable's constraints. A graph is arc-consistent if every variable is arc-consistent with every other variable

2.2   CSP Practice: 4-Queens problem

Consider a 4x4 chess-board. Place four queens on the board such that no two of the queens can attack each other. Queens can attack other queens placed in the same row, column, or along the same diagonal

For example, if we place a queen on (2,2), the following squares can not be occupied by a queen.

Since we know that each column can only have one queen, let us represent this problem with four variables, X_1, X_2, X_3, X_4, where X_i represents the row of the queen in column i. In the example above, $X_2 = 2$. The domain of each X_i is $\{1, 2, 3, 4\}$.
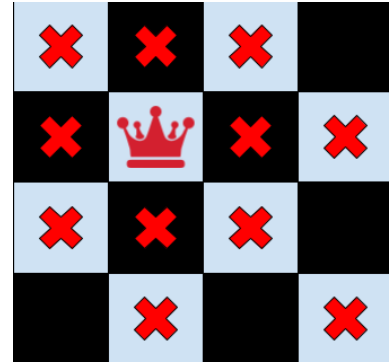
We can represent this using the following constraint graph. Each edge in the graph represents a constraint that the two queens connected by the edge may not be in the same row and that two queens may not be along the same diagonal.
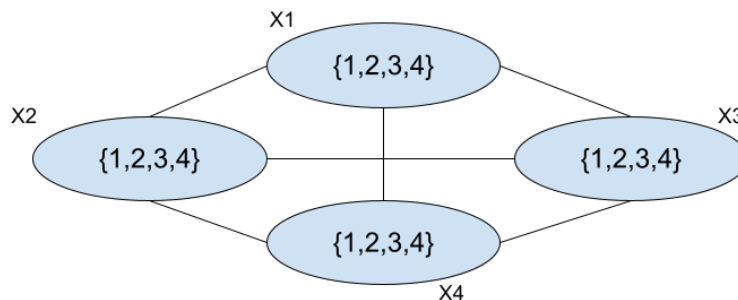
2.2.1   Problem 1

Execute the AC-3 algorithm with the additional constraint that $X_1 = 1$. Does a solution exist? How can you tell?

First, we add all the arcs in the CSP to the queue: $[X_1 - X_2, X_2 - X_1, X_1 - X_3, X_3 - X_1, X_1 - X_4, X_4 - X_1,$ $X_2 - X_3, X_3 - X_2, X_2 - X_4, X_4 - X_2, X_3 - X_4, X_4 - X_3]$. For each arc (X_i, X_j) in the queue, we check whether X_i is consistent with respect to X_j and change the domain of X_i to make it arc-consistent with X_j. If the domain is altered, (X_k, X_i), where X_k represent neighbors of X_i, are added to the queue.

1. $X_1 - X_2$ No changes made
2. $X_2 - X_1$ Remove 1 and 2 from the domain of X_2
3. $X_1 - X_3$ No changes made
4. $X_3 - X_1$ Remove 1 and 3 from the domain of X_3
5. $X_1 - X_4$ No changes made

6. $X_4 - X_1$ Remove 1 and 4 from the domain of X_4

7. $X_2 - X_3$ Remove 3 from the domain of X_2 (If $X_2 = 3$, there is no value in the domain of X_3 that can satisfy the constraints)

8. $X_3 - X_2$ Remove 4 from the domain of X_3

9. $X_2 - X_4$ No changes made

10. $X_4 - X_2$ Remove 2 from the domain of X_4

11. $X_3 - X_4$ Remove 2 from the domain of X_3

Since the domain of 3 is empty, there is no solution to this CSP where $X_1 = 1$

2.2.2 Problem 2

Execute backtracking with forward-checking to find a solution to the above CSP. Assign variables in the order X_1, X_2, X_3, X_4 and the values from 1 to 4.

1. $X_1 = 1$. Forward-checking removes 1 and 2 from the domain of X_2, 1 and 3 from the domain of X_3, 1 and 4 from the domain of X_4

2. $X_2 = 3$. Forward-checking removes 2 and 4 from the domain of X_3. Since the domain of X_3 is now empty, we backtrack

3. $X_2 = 4$. Forward-checking removes 4 from the domain of X_3, 2 and 4 from the domain of X_4

4. $X_3 = 2$. Forward-checking removes 3 from the domain of X_3. Since the domain of X_3 is now empty, we backtrack

5. $X_1 = 2$. Forward-checking removes 1,2, and 3 from the domain of X_2, 2 and 4 from the domain of X_3, and 2 from the domain of X_4

6. $X_2 = 4$. Forward-checking removes 3 from the domain of X_3 and 4 from the domain of X_4

7. $X_3 = 1$. Forward-checking removes 1 from the domain of X_4

8. $X_4 = 3$. Solution is reached where $X_1 = 2$, $X_2 = 4$, $X_3 = 1$, and $X_4 = 3$