Jia-Jia (Jay) Lin
CSE 415, Winter 2020
February 3rd, 2020

# Assignment 4: Worksheet

1. **Blind Search**
    a. A possible state representation is a 2D int array with 6x6 = 36 total elements representing each square in Sudoku.
    b. def successors(s):

```
    list = []
    # loop through all possible choices from this state
    for i in range(7):
        for square in s state:
            if square is Empty: list.append(state with i in Z)
    return list
```

    c. def is_goal(s):

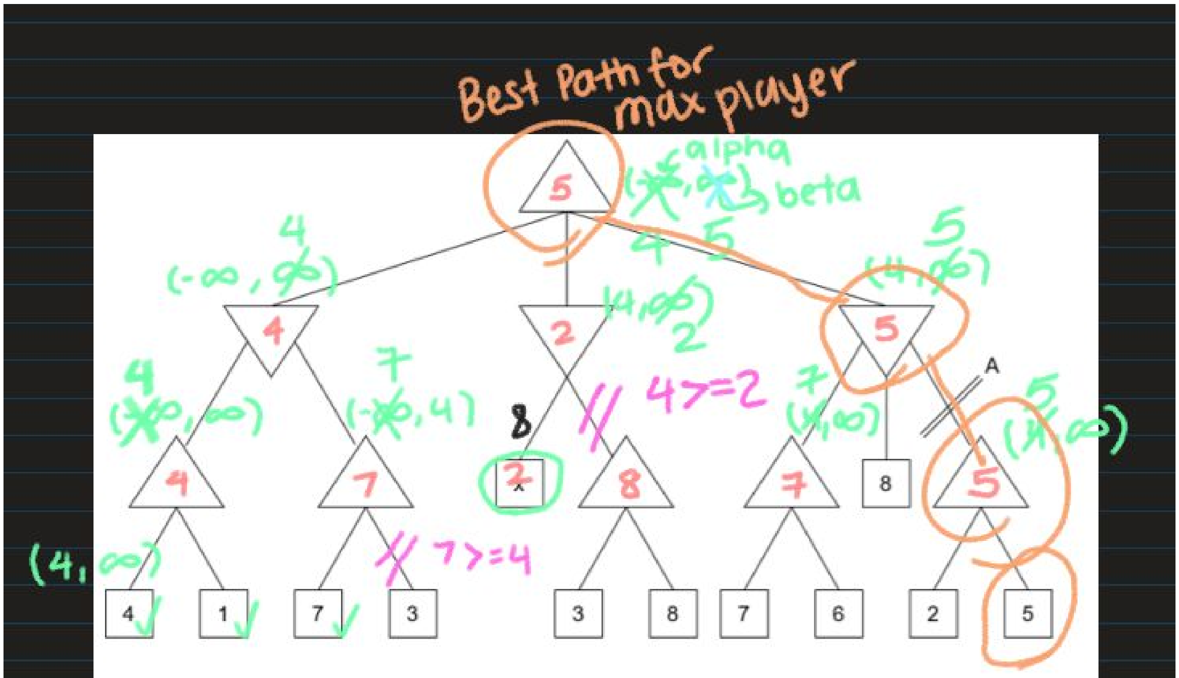
```
    return (s == 2D array of final correct solution)
```

2. **Heuristic Search**
    a. When selecting a heuristic, the challenge is deciding what the value represents, making sure it is reasonable, admissible, and/or consistent. Trade-offs include making the heuristic values too small (low risk of overestimating the distance to the goal node but not very efficient compared to a regular heuristic search) or too big (very efficient but could overshoot the goal node).
    b. h1 and h2 are admissible.
        i. h3 has a heuristic value that could potentially overshoot the actual distance to the goal from Node B (12>11)
    c. None of the heuristics shown above are consistent
        i. h1: s0 to Node B, 14-4=10, which is not <= 3, the cost. h2: 14-10=4 not <= 3, h3: 12-7=5 not <= 3 from Node B to D
        ii. The above shows that the condition to be consistent, that every node’s heuristic value being less than or equal than its successors’ heuristic + cost to get there, is not met.
    d. An A* search algorithm is optimal when using a consistent and admissible heuristic. While none of them are consistent, I would use h2 because it does not overestimate the distance to the goal node, but still has a better efficiency than h1.
    e. Final Solution Path: [(s0,6),(B,8), (D,9), (H,10),(E,12),(gamma,14)]
        i. Path A* traversed: s0 -> B -> A -> G -> D -> F -> H -> E

**Table for Question 2 Part E: A* Search path**

| Current Node | OPEN list | CLOSED list |
|---|---|---|
| s0 | [(B,8),(A,9),(G,9),(F,10),(D,11)] | [s0, 6] |
| B | [(A,9),(G,9), (D,9), (F,10),(C,17)] | [(s0,6),(B,8)] |
| A | [(G,9), (D,9), (F,10),(C,17)] | [(s0,6),(B,8)] |
| G | [(D,9), (F,10),(H,15),(C,17)] | [(s0,6),(B,8), (G,9)] |
| D | [(F,10),(H,10),(C,17)] | [(s0,6),(B,8), (D,9)] |
| F | [(H,10),(C,17)] | [(s0,6),(B,8),(D,9)] |
| H | [(E,12),(C,17),(gamma, 20)] | [(s0,6),(B,8), (D,9), (H,10)] |
| E | [(gamma,20), (C,17)] | [(s0,6),(B,8), (D,9),(F,10),(H,10),(E,12)] |
| gamma | [(C,17)] | [(s0,6),(B,8), (D,9), (H,10),(E,12),(gamma, 14)] |

3. **Adversarial Search** *(note: red pen = minimax, green pen = alpha-beta pruning)*
    a. (part a and b drawing shown below)



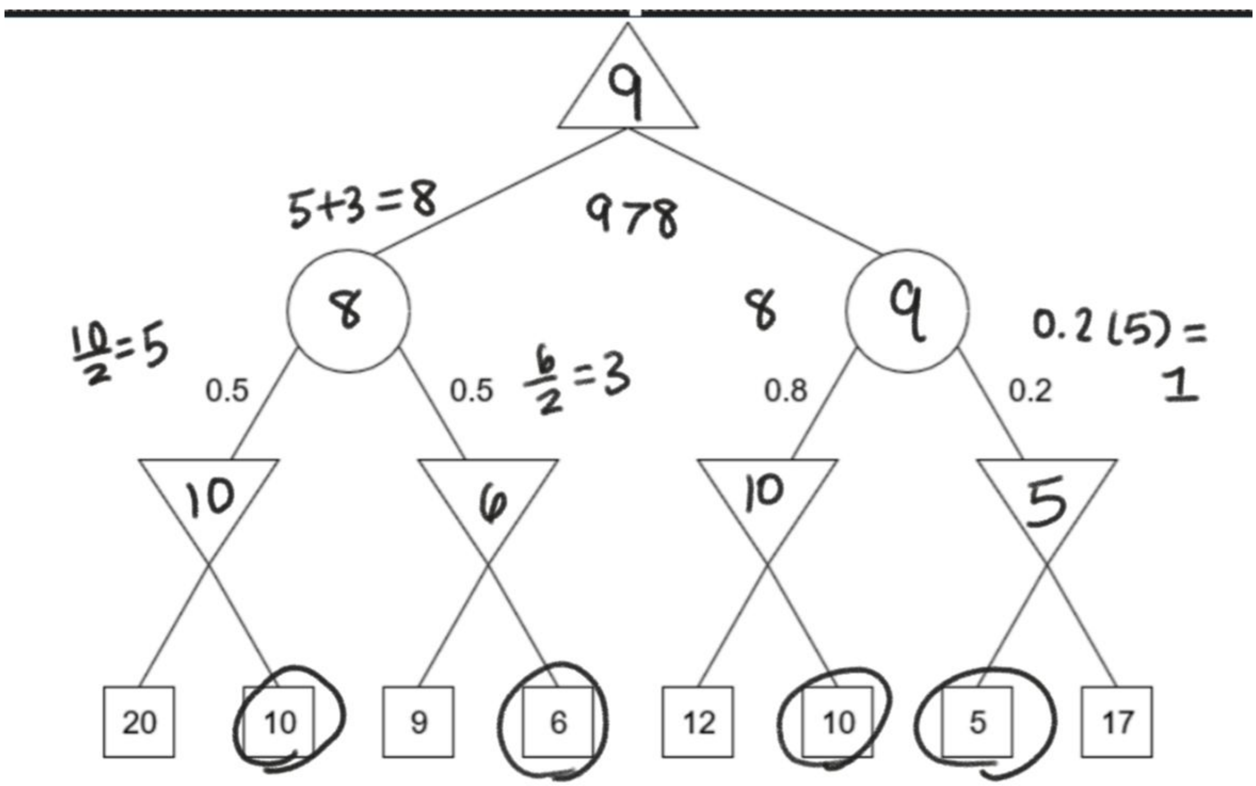

(a) (5 points) Let $x = 2$ in the static evaluation node containing the variable in the tree. Fill in the nodes in the tree with the correct values selected by the maximizing and minimizing players during the minimax algorithm.

(b) (5 points) Let $x = 2$ in the static evaluation node containing the variable in the tree. Apply alpha-beta pruning to the tree. Specifically, mark the appropriate edges in the tree with a double slash (//) to indicate which nodes were pruned using the algorithm.

c. What is the smallest value of x where A is pruned using alpha-beta?

X = 7, *because the minimizing player will choose 7 in the middle branch, which then gets passed to the right side of the tree later. At the right child, it will be [4,7] and goes to its left child first, which is the maximizing player (chooses 7) → [7,7]. 7 is greater than or equal to 7, so the remaining branches are pruned.*

d. Consider the following expectimax game tree. (Answers in drawing below)



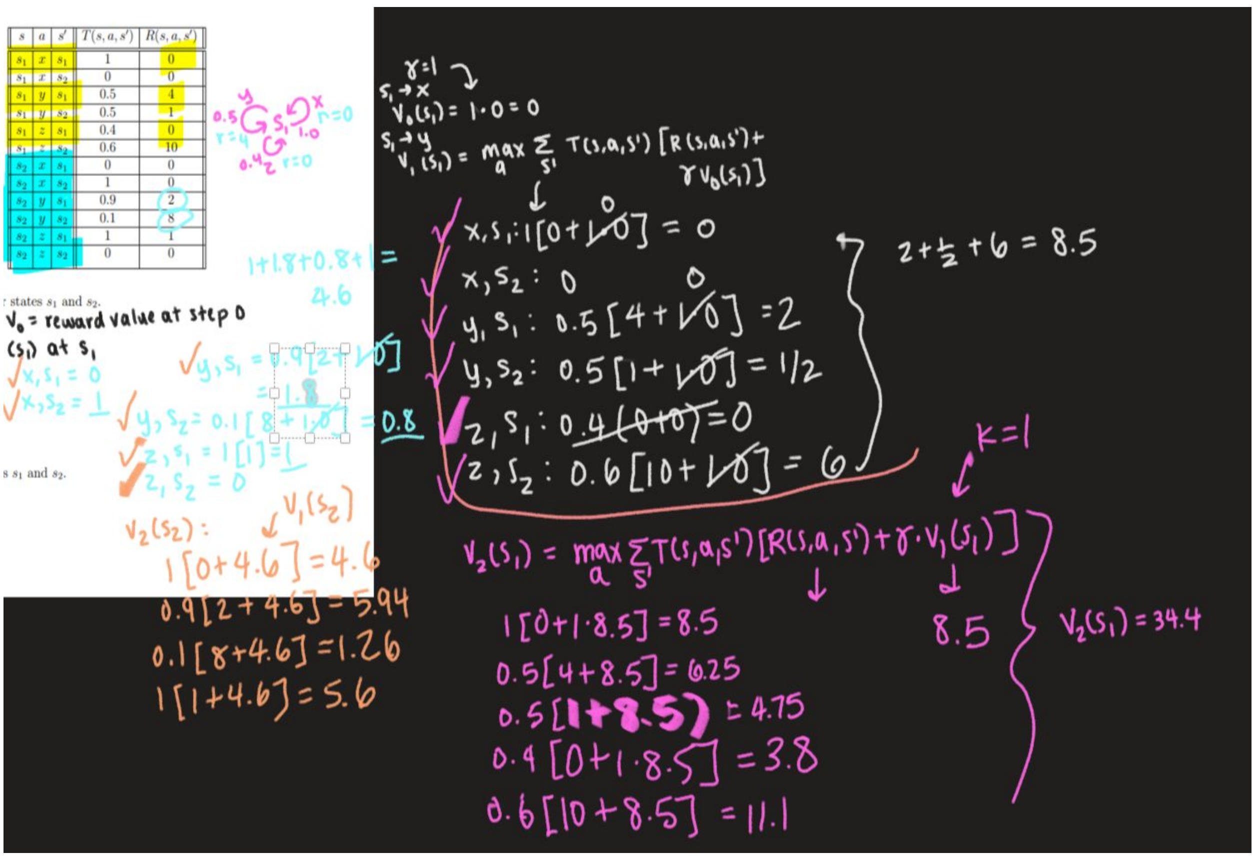
4. **Markov Decision Processes**
    a. Actions and states
        i. Actions for this MDP: a1 = Roll 1, a2 = 3, a3 = 5, or a4 = 6, or a0 = STOP
        ii. 8 total states: s0 = Initial state, sums = {s1,s2,s3,s4,s5,s6}, s7 = Final State
    b. Full transition function
        i. T(s,a,s’) = 0.25, probability of rolling a number on a tetrahedral dice; given s = s0
            1. a is within {a1,a2,a3…} and NOT a0
            2. s’ is within {s1,s2,s3…} and NOT s7
        ii. Example: T(s0,a1,s1) = 0.25
    c. Reward functions
        i. R(s0,a2,s3) = 3 (total sum of 3)
        ii. R(s0,a3,s5) = 5
        iii. R(s0,a4,s6) = 6
        iv. R(s1,a1,s2) = 2
        v. R(s1,a2,s4) = 4
        vi. R(s1,a3,s6) = 6
        vii. **R(s,a,s’) = -10000 where s’ = s7 and both s = s0 and a = a0, OR a is not a0**
            1. In words: The reward is very bad if the player immediately forfeited the game at the first turn or if they got a bust.
            2. Example: R(s0,a0,s7) = -10000
        viii. **R(s,a,s’) = 1, where s’ = s1**
            1. Example: R(s0,a1,s1) = 1
        ix. **R(s,a,s’) = 2, where s’ = s2**
            1. Example: R(s1,a1,s2) = 2
        x. R(s,a,s’) = 3, where s’ = s3
        xi. R(s,a,s’) = 4, where s’ = s4
        xii. R(s,a,s’) = 5, where s’ = s5
        **xiii.** **R(s,a,s’) = 10000, where s’ = s6**

1. The maximum / best possible score

d. The optimal policy is one where the agent plays carefully but reasonably: avoids stopping the game too early and not overshooting the number 7. The agent should have an increasingly higher preference for stopping when reaching a total sum close to 7.

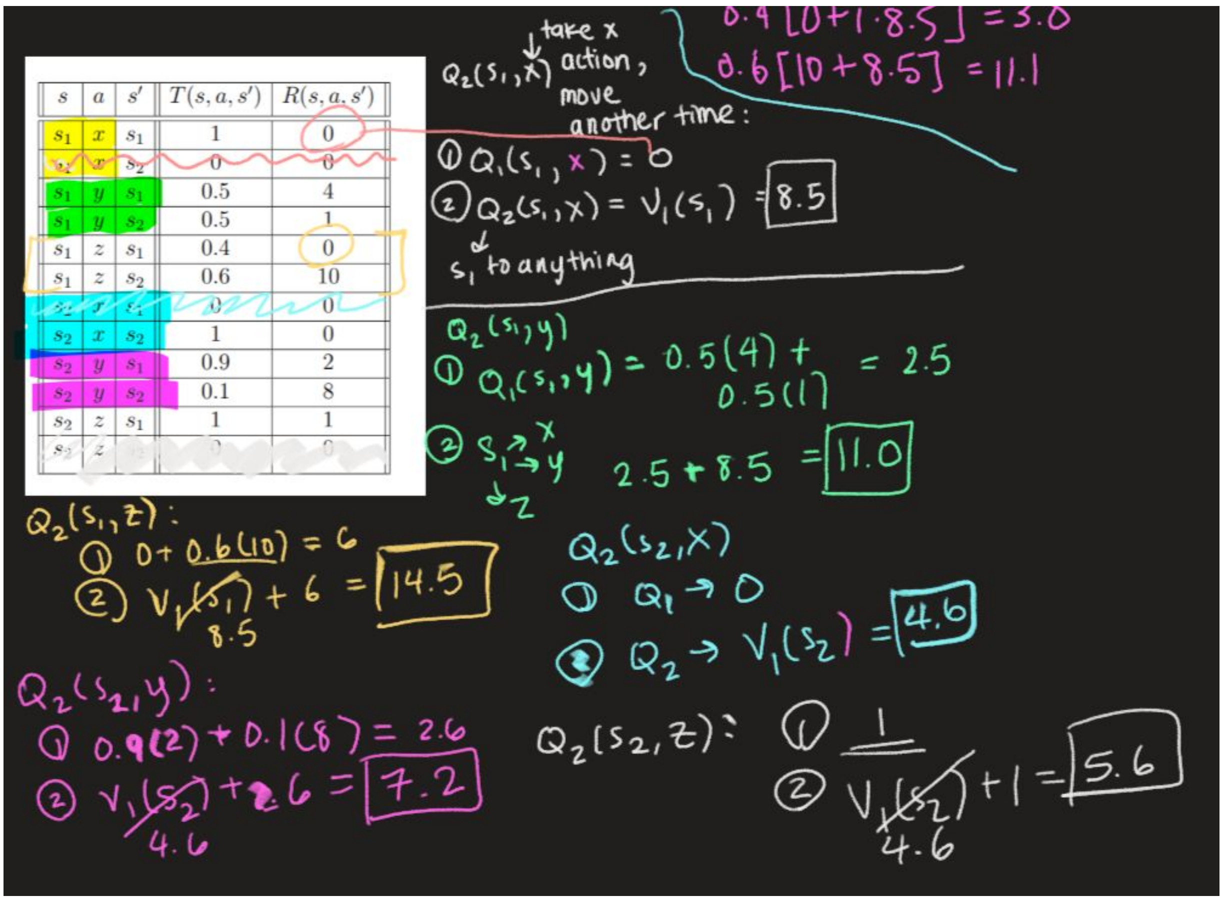
5. **Computing MDP State Values and Q-Values**
    a. 0
    b. 0
    c. 8.5
    d. 4.6
    e. 34.4
    f. 17.4

(see work for value iteration below)



g. (Question 5 continued) 8.5
h. 11.0
i. 14.5
j. 4.6
k. 7.2
l. 5.6

(see work for Q values below)



**Finally, compute V2 and Q2 for both states, but with γ = 0.5.**

m. 29.75
n. 15.1
o. 8.5
p. 11.0
q. 14.5
r. 4.6
s. 7.2
t. 5.6

(see work below)

- Reasoning for Q-values
  - Since Q0 = 0, the Q2 values didn't change regardless of gamma value because it is (gamma*0 + the reward)*probability for the first step and then add V1(s) which still involved multiplying gamma by 0 since V0 = 0

Finally, compute $V_2$ and $Q_2$ for both states, but with $\gamma = 0.5$.

(m). $V_2(s_1) =$ 29.75

(n). $V_2(s_2) =$ 15.1

$V_2(s_2) | \gamma = 1 = 4.6$

$\gamma = 1$ 2

$V_1(s_1) = 8.5$ because $V_0 = 0$

$4.25 + 4.125 + 2.625 + 1.7 + 8.55 = 21.25$

$2.3 + 3.87 + 1.03 + 3.3 = 10.5$

(o). $Q_2(s_1, x) =$ 8.5

(p). $Q_2(s_1, y) =$ 11.0

(q). $Q_2(s_1, z) =$ 14.5

(r). $Q_2(s_2, x) =$ 4.6

(s). $Q_2(s_2, y) =$ 7.2

(t). $Q_2(s_2, z) =$ 5.6