CSE 415 Autumn 2023 Assignment 4

Students: Please put your name and all your solutions in blue.

Last name: Yamashita First name: Shinji

This is an individual-work assignment. Do not collaborate on this assignment.

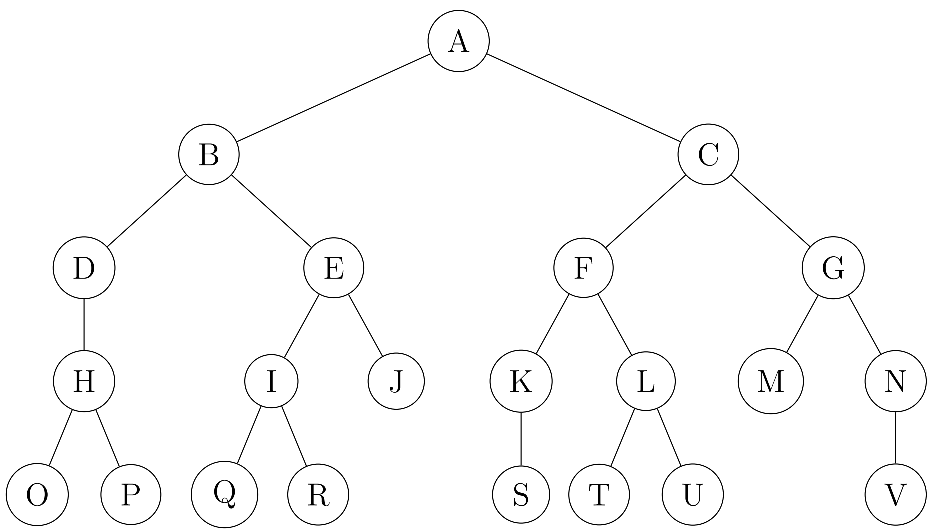
Prepare your answers in a neat, easy-to-read PDF. Our grading rubric will be set up such that when a question is not easily readable or not correctly tagged or with pages repeated or out of order, then points will be be deducted. However, if all answers are clearly presented, in proper order, and tagged correctly when submitted to Gradescope, we will award a 5-point bonus.

If you choose to choose to typeset your answers in Latex using the template file for this document, please put your answers in blue while leaving the original text black.

---

1 Basic Search

Use the following tree to answer the questions below comparing Breadth First Search, Depth First Search, and Iterative-Deepening Depth First Search. Assume that children are visited from left to right.



(a) (3 points) Write out the order that the nodes are expanded in using Breadth First Search, starting from A and searching to K.

A→B→C→D→E→F→G→H→I→J→K

(b) (3 points) Write out the order that the nodes are expanded in using Depth First Search, starting from A and searching to K.

A→B→D→H→O→P→E→I→Q→R→J→C→F→K

(c) (3 points) Write out the order that the nodes are expanded in using Iterative-Deepening Depth First Search, starting from A and searching to K. If a node is repeated, make sure to include it each time it is expanded.

A→A→B→C→A→B→D→E→C→F→G→A→B→D→H→E→I→J→C→F→K

(d) (4 points) Which of the three search algorithms (BFS, DFS, IDDFS) has the smallest maximum size of the open list while searching from A to K? What is the maximum size of the open list for that algorithm?

IDDFS has the smallest maximum size of the open list while searching from A to K. The maximum size of the open list is 3 for IDDFS

(e) (2 points) True or False: BFS, DFS, and IDDFS will each return the same path starting at A going to K.

True

(f) (2 points) True or False: Given the same start and goal state, BFS, DFS, and IDDFS will always return the same path for any search graph.

False

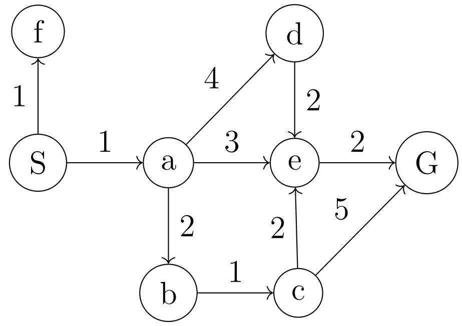
(g) (4 points) Under what conditions might BFS be a better choice of search algorithms when compared to DFS? (Give one possible reason, 1 sentence)

BFS would be a better choice of search algorithms when the goal state is at a shallow depth close to the root node and in the subtree that isn't intially traversed, usually the right

(h) (4 points) Under what conditions might IDDFS be a better choice of search algorithms when compared to BFS? (Give one possible reason, 1 sentence)

When the goal state is deep in the tree and also in the subtree that is intially traversed, adjacent to the root node usually the left side.

2 Heuristic Search



For the following questions, consider three heuristics h_1, h_2, h_3. The table below indicates the estimated cost to goal, G, for each of the heuristics for each node in the search graph.

state (s)	S	f	a	b	c	d	e	G
heuristic $h_1(s)$	6	12	5	2	3	4	2	0
heuristic $h_2(s)$	6	5	6	4	4	4	1	0
heuristic $h_3(s)$	5	8	5	4	3	3	2	0

(a) (2 point) What does it mean for a heuristic to be "admissible"?

It is admissible when if it never overestimates the true distance from the start state to the end goal

(b) (3 points) Which heuristics among $\{h_1, h_2, h_3\}$ shown above are admissible? Identify any violation(s) found in making this determination.

$h_1(s)$, $h_3(s)$ are admissible
$h_2(s)$ is not admissible because the actual shortest path from a to g is a cost of 5 and then heuristic is 6 at a which overestimates the true distance

(c) (2 point) What does it mean for a heuristic to be "consistent"?

A heuristic is consistent when the goal state of the heuristic is 0 and for every edge (x,y) in the graph $h_k(x)$ - $h_k(y)$ $\leq$w(x,y)

(d) (3 points) Among the heuristics you identified as admissible in part (a), which are also consistent? Identify any violations found in making this determination.

Only $h_3(s)$ is consistent since $h_1(s)$ violates consistency as the combined difference cost of heuristic from edge a going to b is three which is larger than the actual path of the edge which is two.

(e) (4 points) Show the node expansion order going from S to G using $A*$ with h_1. Also provide the final path obtained.

S→a→b→e→G
Path: S→a→e→G

(f) (4 points) Show the node expansion order using $A*$ for h_3.

S→a→e→G

(g) (2 points) Of the heuristics you used in (e) and (f) above, which would you consider the better heuristic? Explain why, using examples from your attempts to use each heuristic to support your explanation.

$h_3(s)$ is the better heuristic because you expand one less node than $h_1(s)$ and the solution path is the same

(h) (2 points) What does it mean for one heuristic to dominate another heuristic?

For all states, $h_i(s)$ ≥h$_j(s)$ assuming both heuristics are admissible. It also means that it is more informed

(i) (1 points) Consider h_1 – what change(s) would you make to this heuristic to make it the dominant heuristic of the three options available? Describe both the change you'd make and how the heuristic now satisfies the conditions for dominance.

I would set a=7, b=5, c=4 so that now every heuristic for each state in $h_1(s)$ is larger than the same state for the other two heuristics.

(j) (2 points) Explain why, when using the $A*$ search algorithm, it is important to continue the expansion process until the goal state is removed from the OPEN list and becomes the current state, rather than terminating as soon as the GOAL sate is found.

It is important to continue the expansion process until the GOAL state is the first state in the priority queue and popped off since there could potentially be another short path with a really low cost to the GOAL state. If you terminate when the GOAL state is found, you might not get the shortest path.

3 Designing Heuristic Functions

As the midterm season is coming up, you as the most senior student want to create an informative welcome video for the incoming first-year to explain about how to navigate the school. Based on your many years experience as a student, you have realized that new students do not like to listen to a long talk given by a single person. However, as a student studying computer science, you are too lazy to create a video yourself. So, you have come up with an idea that you will instead be just combining video footage from previous years' recordings.

More formally, this exercise looks into the use of state-space search to design an *optimal* video mix. Suppose there is a dataset $V = \{v_1, v_2, \ldots, v_n\}$ of video segments, together with information about segment lengths in seconds and the topics covered. The *optimal* video is defined to be a set of of video segments that cover the set of specific topics while still using the least total of segment lengths. Note that the order of video segments does not affect the optimal conditions.

Segment Name	**Segment Length (seconds)**	**Topics covered**
`2023au`	10	`[welcome]`
`2023su`	30	`[skiing, views]`
`2023sp`	50	`[welcome, AI, ChatGPT]`
`2023wi`	40	`[salmon, dragons, skiing]`
`2022au`	50	`[skiing, ChatGPT]`

Table 1: Dataset of video segments

For simplicity, we formulate this problem as a graph search by representing states (and a node for each state) as $s_i = (r_i, m_i)$ where r_i is the list of remaining topics to be covered and m_i represents the mix so far: a list of video segments that have been added. Note that none of these video segments in m_i has any topics in r_i, since those topics are the ones that still need to be covered by some additional video segment.

The set of operators $\Phi = \{\phi_1, \phi_2, \ldots, \phi_n\}$ has one operator for each video segment; the operator ϕ_j means to add the j^{th} video segment to the mix. The precondition for ϕ_j is that the j^{th} video segment covers (among other possible topics) the first element of r_i. The state-transformation for ϕ_j is to remove any topics from r_i that are covered by the j^{th} video segment and to add this video segment to the mix m_i.

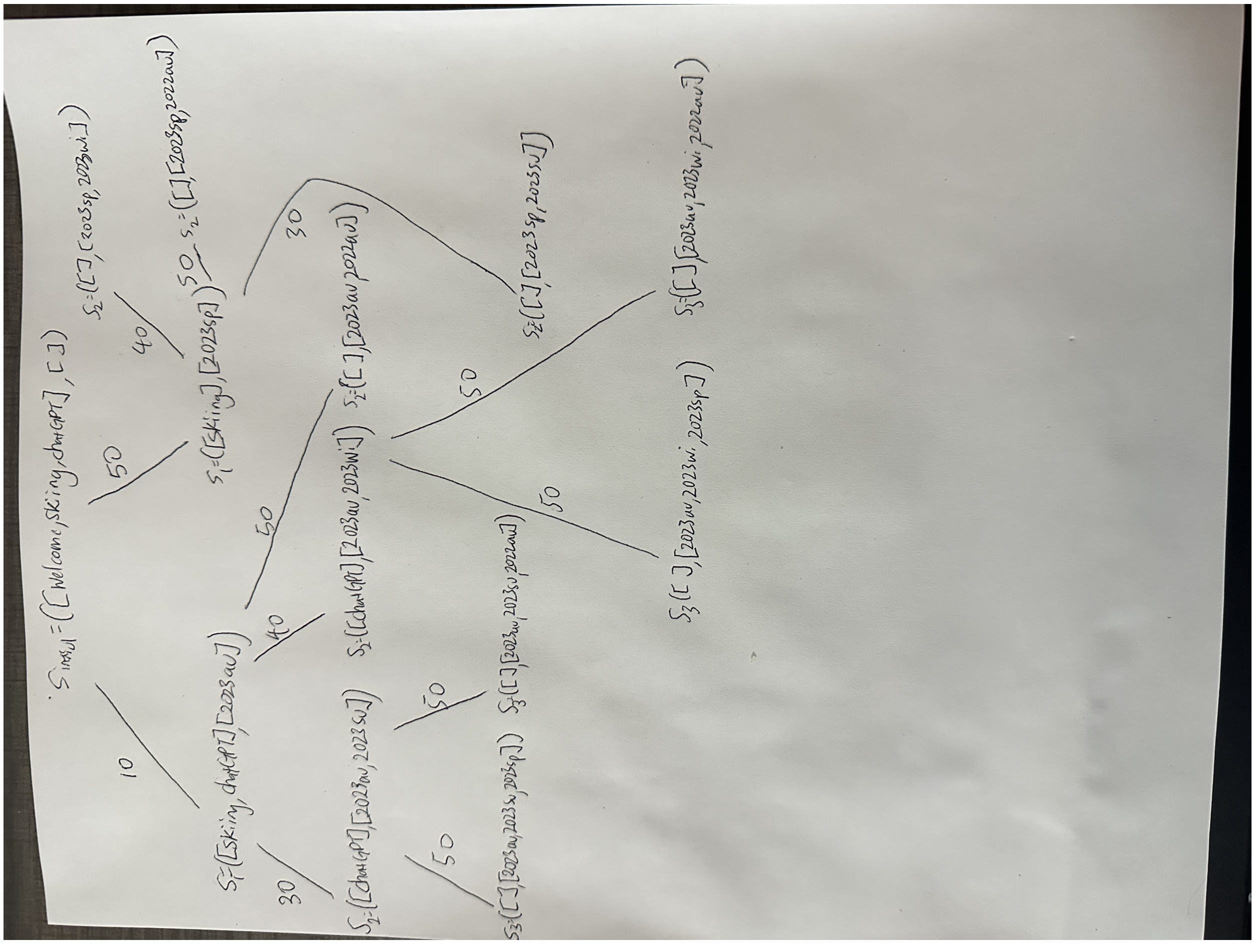


Figure 1: 12 total states reachable from initial state

Thus, for any state $s_i = (r_i, m_i)$, its successors are found by considering all video segments not in m_i that contain the first topic from r_i. For instance, the successors of state ([welcome, dragons], []) are ([dragons], [2023au]) and ([dragons], [2023sp]). The weight of the edge from s_i to the successor produced by ϕ_j is defined to be the segment length of the j^{th} video segment.

(a) (10 pts) For this assignment, assume the initial state is
$s_{\texttt{initial}} =$ ([welcome, skiing, ChatGPT], []).
Draw the portion of the search space reachable from this state.

(b) (5 pts) Give a non-trivial admissible heuristic function $h(s)$. (This could then be used with an A* search for this problem, and in theory, much larger examples of this sort of problem.) Note that $h(s) = 0$ is the trivial heuristic function.

My heuristic function takes the minimum segment length that represents the video segment that can cover at least the number of remaining topics. So for example, ([skiing, chatGPT]) would have the heuristic value of 30 since there are four video segments that can cover at least the number of remaining topics which is two, since 2023su has two topics, 2023sp has three, 2023wi has three, and 2022au has two. In this case 2023su would have the minimum segment length of 30.

(c) (5 pts) Prove that your heuristic function is admissible.

My heuristic function would be admissible since the shortest true distance would always the minimum of the segment lengths so in the worst case, my heuristic function would only be equal to but never greater than the actual true distance because it is a lower bound.

(d) (5 pts) Consider the following heuristic function. Is the heuristic function admissible? If yes, show the proof. Otherwise, please provide one or more counterexamples.

$$h(s) = h\left((r, m)\right) = \sum_{t \in r} q(t)$$

where $q(t)$ is the length of the smallest video segment that covers topic t.

The heuristic function is not admissible because the true path of the cost from the initial state to the closest goal state ([welcome, skiing, chatGPT], []) is 60 and the heuristic function for the initial state would be 90 since the length of the smallest video segment for welcome is 10, skiiing is 30, and chatgpt is 50 so 10+30+50 would equal to 90 as it is a summation. This means the heuristic is larger than the actual path, making it not admissible.

4 Adversarial Search: Static Evaluation and Minimax

For this problem, we will be referring to the paper "Some Studies in Machine Learning Using the Game of Checkers"[1] written by Arthur L. Samuel at IBM in 1959. It is one of the pioneering works not only for adversarial search, but also for modern machine learning algorithms.

Please read the 'Introduction' and 'The basic checker-playing program' sections. Then read the following (sub)sections to answer the questions below: 'Ply limitations,' 'Rote Learning and its variants,' 'Learning procedure involving generalizations' (first two subsections). The number of the relevant page for each question is included in brackets after the question.

4.1 Warm up

(a) T/F The program evaluates a board position at a minimum look-ahead ply if the last move was a jump (p538).

False

(b) Why is it not advisable to take the initial move which leads to the highest scoring board position (p538)?

It is not advisable to take the initial move which leads to the highest scoring board position since to reach this position it would require cooperation of the opponent.

(c) T/F The most important part of the scoring polynomial used during play is the 'inability for one side or the other to move' (p536).

True

4.2 Learning

(d) In machine learning algorithms, it is common to use "regularization", an approach that leads your learner to (i) ignore some information, in order to (ii) prioritize more generalizable concepts from your training data. Which two terms in the paper allude to (i) and (ii)? Explain why they are beneficial to the learning program (p540).

Cataloging and Culling refer to these tow terms. They are beneficial to the learning program to minimize storage requirements and to reduce search time as it allows the algorithm to search for moves that lead to winning combinations instead of unnecessary moves that just lead to an increase in time complexity

[1](link to the paper)

(e) When attempting to generalize a game learner, a subset of coefficients from the scoring polynomial are selected for the static evaluation. However, this can sometimes lead to severely under-performing agents as can be seen by, for example, losing three turns in a row. A suggestion for such a case is to set which polynomial term to zero and why (p542)?

It reduces the coefficient of the leading term of alpha to zero because it was trapped since it found a secondary maxima in a multi-dimensional scoring system leading it to lose the turns so that it can improve by learning to ignore the most important feature.

4.3 Efficiency

(f) What is one way to increase the number of backed-up plies used in selecting the next move, rather than what is typically possible without incurring much extra computation time cost (p540)?

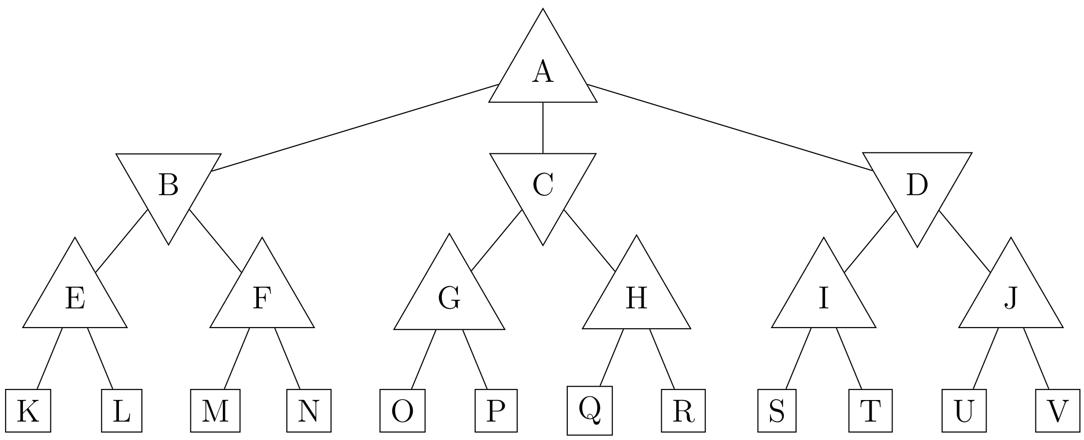
One way is simply by decreasing the magnitude of the evaluation score a small amount each time it is backed up a ply level during the analyses. This allows the agent to choose a low ply alternative if winning and high ply alternative if losing.

(g) Why are the records of board positions arranged such that those likely to be seen at the beginning of the game appear earlier on tape (p540)?

This is because they are most likely to be needed during the course of the game since the board positions at appear at the beginning of the games are more frequent.

5 Adversarial Search and Alpha-Beta Pruning

Use this search tree along with the table of state evaluation values to answer the questions below. By default, children are processed from left to right.



state	K	L	M	N	O	P	Q	R	S	T	U	V
evaluation	3	7	1	-5	4	8	-1	-3	-4	-2	6	2

(a) (10 points) Use minimax to perform adversarial search with alpha-beta pruning on the tree above. Fill in the values in the table below as you go. For α and β values, write the values that are passed from the state's parent to that state. For example, the value of state K will not be shown in the $\alpha - \beta$ values of state E, but would be reflected in the $\alpha - \beta$ values of state L. If a state does not have to be evaluated, do not write any values for it in the table.

state	value	α	β
A	1	N/A	N/A
B	1	-∞	∞
C	-1	1	∞
D	-2	1	∞
E	7	-∞	∞
F	1	-∞	7
G	8	1	∞
H	-1	1	8
I	-2	1	∞
J			
K	3	-∞	∞
L	7	3	∞
M	1	-∞	7
N	-5	1	7
O	4	1	∞
P	8	4	∞
Q	-1	1	8
R	-3	1	8
S	-4	1	∞
T	-2	1	∞
U	6		
V	2		

(b) (15 points) Consider optimizing alpha-beta search on this tree by re-ordering the nodes of the tree. By changing the order of the children of some nodes, the algorithm may need to evaluate more or less states due to alpha-beta pruning. What nodes can be re-ordered within the tree to minimize the number of states that need to be evaluated? How many fewer states need to be evaluated when compared to the original ordering?

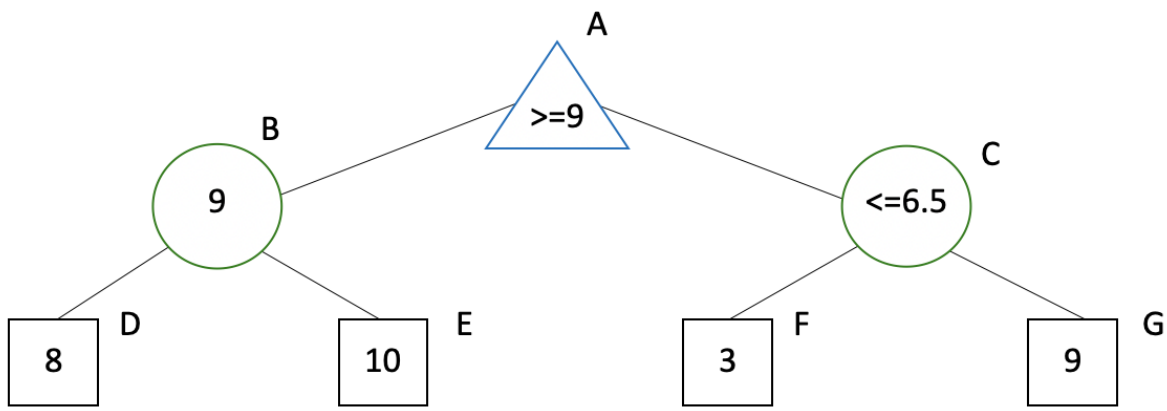
When re-ordering, make sure that all parent-child relationships in the tree are unchanged, and only the order of a node's children are changed.

Hint: our staff solution makes just two changes to the ordering.

Switch G and H nodes so that H sub tree isn't traversed meaning H,Q,R is not traversed as -1 is going to be less than C's alpha which is 1. We also switch E and F nodes so that the node N isn't traversed, making it 4 fewer states to be evaluated compared to the original ordering.

6 Pruning with Chance Nodes

Although alpha-beta pruning cannot be applied directly to searching trees that contain chance nodes, reasoning like that inherent in alpha-beta search can sometimes be applied when static values are constrained to lie within given ranges of values. For example, if all leaf-node values must be values $f(s)$ such that $0 \leq f(s) \leq 10$, then node G can be pruned in the tree below, because Max can get 9 by moving left at A to B, and if Max goes to C and finds $f(F) = 3$, Max can reason that getting $f(G)$ is useless because if $f(G) = 10$ which is larges allowed, then the value at C is (3+10)/2 = 6.5, which is inferior to 9.



(10 points) In the following game tree, determine where pruning can be performed using the same range assumption as above. Show where there are cutoffs

(15 points) Explain your reasoning for each cutoff.

The first cutoff happens when the leaf node is discovered to be 10, then we know that since the range of leaf nodes has to be less or equal to ten, it is redundant to search for right side of the child node as it can only be smaller or same. Then, since root node is minimizing, we know that the average has to be greater than or equal to 10+0/2=5, which is greater than the 3 so we can prune this side of the tree.

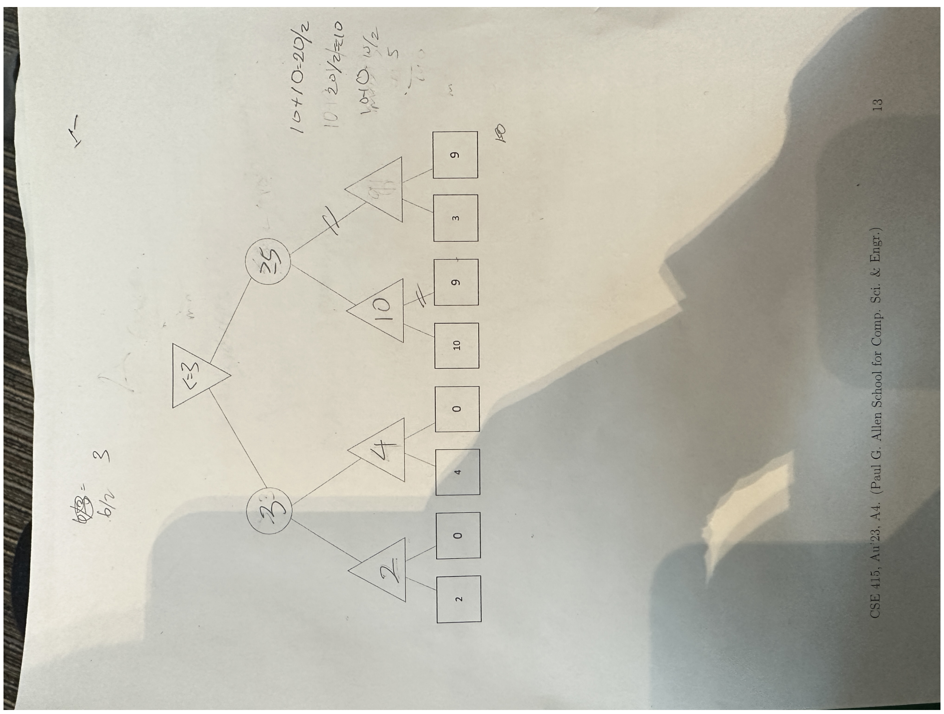


Figure 2: Game tree with pruning