CS 540: Introduction to Artificial Intelligence
Homework Assignment # 5

Assigned: 10/9
Due: 10/16 8:50am

Hand in your homework:

If a homework has programming questions, please hand in the Java program. If a homework has written questions, please hand in a PDF file. Regardless, please zip all your files into hwX.zip where X is the homework number. Go to UW Canvas, choose your CS540 course, choose Assignment, click on Homework X: this is where you submit your zip file.

Late Policy:

All assignments are due at the beginning of class on the due date. One (1) day late, defined as a 24-hour period from the deadline (weekday or weekend), will result in 10% of the total points for the assignment deducted. So, for example, if a 100-point assignment is due on a Wednesday 9:30 a.m., and it is handed in between Wednesday 9:30 a.m. and Thursday 9:30 a.m., 10 points will be deducted. Two (2) days late, 25% off; three (3) days late, 50% off. No homework can be turned in more than three (3) days late. Written questions and program submission have the same deadline.

Assignment grading questions must be raised with the instructor within one week after the assignment is returned.

Collaboration Policy:

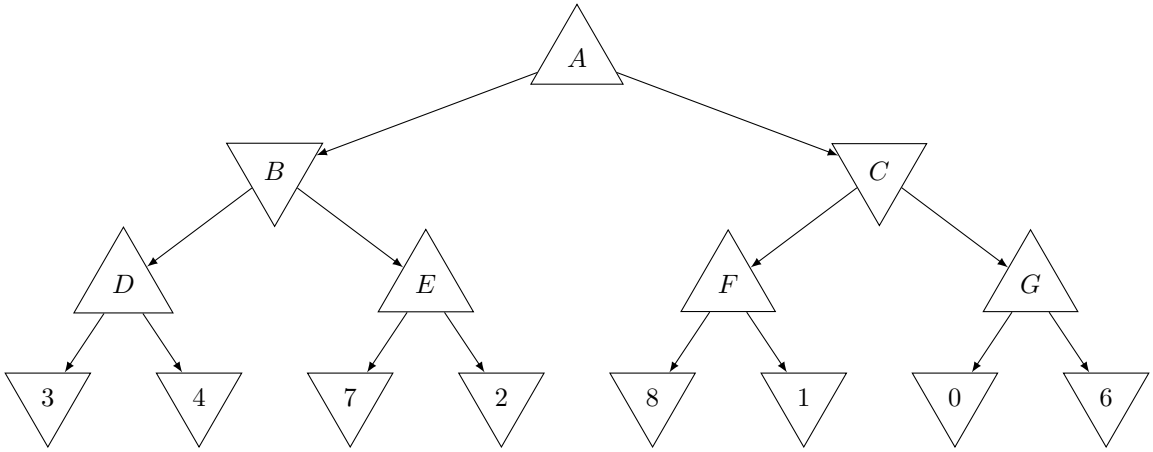
You are to complete this assignment individually. However, you are encouraged to discuss the general algorithms and ideas with classmates, TAs, and instructor in order to help you answer the questions. You are also welcome to give each other examples that are not on the assignment in order to demonstrate how to solve problems. But we require you to:

- not explicitly tell each other the answers
- not to copy answers or code fragments from anyone or anywhere
- not to allow your answers to be copied
- not to get any code on the Web

In those cases where you work with one or more other people on the general discussion of the assignment and surrounding topics, we suggest that you specifically record on the assignment the names of the people you were in discussion with.

Question 1: Game Tree Search [60 points]

Consider the game tree below. Let Δ and ∇ nodes represent nodes belonging to the maximizing and minimizing player respectively.



1. (20 points) Suppose we wish to run the **Minimax** algorithm on this game tree. Provide the game theoretic values of nodes $A \ldots G$ through optimal play.

2. (10 points) Use alpha-beta pruning to compute the Minimax value at each node for the original game tree (i.e., deterministic environment). Assume that nodes are visited from left to right. In addition to the Minimax values, show your alpha and beta values at each node after the final time it is visited.

3. (5 points) Which branches, if any, in the game tree are pruned? Show this on the graph above.

4. (5 points) Explain, in English, why we would want to use alpha-beta pruning.

5. (5 points) Now suppose we are in a non-deterministic environment with rules as follows. When a player begins their turn they first flip a fair coin (heads probability .5) and if it turns up heads they choose the optimal action. However, if the coin turns up tails they then flip another fair coin to randomly select their action. In this game, what is the probability of a player choosing an optimal action *at each node*?

6. (5 points) Given the game described in part 5, redraw the game tree as follows. For each non-terminal node X, introduce a new chance node X'. Let X' be the only successor of X, while the original successors of X are now given to X'. Label the probability on each edge from X' to its successors. These probabilities should sum to 1 for each X'.

7. (10 points) Using the new game tree in part 6, solve the non-deterministic game. Specifically, provide the *expected* game theoretic value of nodes $A \ldots G$ for the new tree. Round your answers to the nearest 2 decimal places (i.e. $X.XX$). Hint: this is explained on slide p.58.

Question 2: Natural Language Processing [40 points, 5 points each]

This question will require you to do some minimum coding or scripting to get the answers, but it is *not* a programming question. Do not turn in any code. Instead, simply put your answers in the same pdf file as the other questions. You may use any programming language or tools.

Download the zip file `http://pages.cs.wisc.edu/~jerryzhu/cs540/handouts/WARC201709.zip`. Unzip it on your computer, you should see 363 text files. These are your essays from homework 1. We call the whole collection the corpus.

1. Briefly describe in English, the most convenient way you can think of for a computer program to break the corpus into "computer words." For example, if you use Java you may consider Scanner; if you use unix commandline you may use wc. We want you to use the simplest method that you can think of. It is OK if these computer words do not fully agree with our natural language definition of words: For example, you may have a computer word like `However,` with that comma attached. Your method is a crude "natural language tokenizer." When we mention "word" we mean your computer word.

2. Under your method, how many computer word tokens (occurrences) are there in the corpus? How many computer word types (distinct words) are there in the corpus?

3. Sort the word types by their counts in the corpus, from large to small. List the top 20 word types and their counts.

4. Pick 20 bottom word types (they should all have count 1) and list them. You may want to include some strange ones, if any.

5. Let the word type with the largest count be rank 1, the word type with the second largest count be rank 2, and so on. If multiple word types have the same count, you may break the rank tie arbitrarily. This produces $(r_1, c_1), \ldots, (r_n, c_n)$ where r_i is the rank of the ith word type, and c_i is the corresponding count of that word type in the corpus; n is the number of word types. Plot r on the x-axis and c on the y-axis, namely each (r_i, c_i) is a point in that 2D space (you can choose to connect the points or not).

6. Plot $\log(r)$ on the x-axis and $\log(c)$ on the y-axis. You may choose the base.

7. Briefly explain what the shape of the two curves mean.

8. Discuss *two* potential major issues with your computer words, if one wants to use them for natural language processing.