HW2: CSPs and Games

CS4300: Artificial Intelligence **Alan Kuntz**
University of Utah

1 Constraint Satisfaction

After Alice gets back from her ski trip, she decides that she want to plan out her course load over the next 4 semesters. She wants to take AI, and determines that there are six classes that she must take: CS2100 (A), CS3130 (B), CS3500 (C), CS3505 (D), CS4150 (E), and CS4300 (F).

Some of the classes Alice wants to take have prerequisites:

- She must take CS3500 *before* CS3505.
- She must take CS2100 and CS3500 *before* CS4150.
- She must take CS3130, CS3505, and CS4150 *before* CS4300

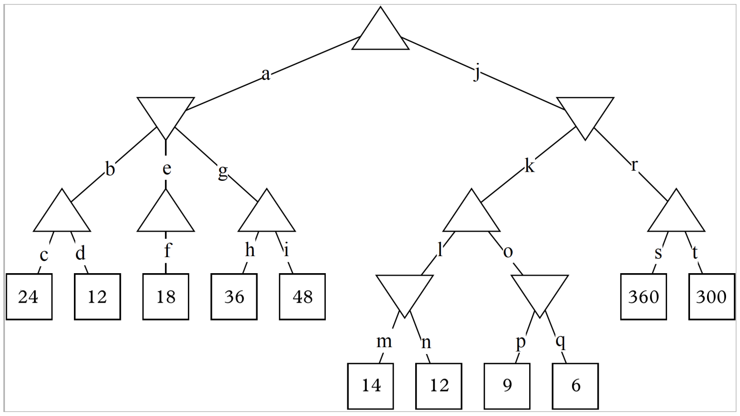
Additionally, she can only take two classes in any given semester.

1. If we use the classes as our variables, what would the full possible domain of each variable be? Do not apply *any* of the constraints.

2. Provide the constraints for the search problem as mathematical expressions. Use the variable names provided after the class names to refer to each class.

3. Give one reason that we might not choose to use semesters as our variables.

2 Adversarial Games



1. Perform minimax search on the given tree. What value is returned (assigned to the root node)?
2. What path is selected by the minimax search you performed in the previous question?

   List the edges in alphabetical order.
3. Perform alpha-beta pruning on the given tree. What edges are pruned?

   You do not need to include an edge if it is the child of an edge that has already been pruned. For example, if **l** is pruned you do not need to list **m** or **n**.

   Again, list the edges in alphabetical order. Traverse branches from left to right.

3 Non-Zero-Sum Games

Alice is taking a class taught by Bob called "Artificial Intelligence." Bob has three ways he can teach the class: "Hard," "Medium" or "Easy." Alice has three ways she can take the class: "Hard Working," "Working" and "Hardly Working." For each of them, there are pros and cons. This gives rise to the following table of rewards. These are written as (A, B) where A is Alice's reward and B is Bob's reward:

	Hard	**Medium**	**Easy**
Hard	$(11, 11)$	$(7, 6)$	$(2, 1)$
Working	$(6, 8)$	$(9, 7)$	$(5, 3)$
Hardly	$(3, 1)$	$(6, 2)$	$(5, 5)$

1. If Bob assumes that Alice will optimize her own reward (i.e., Bob assumes Alice is an optimal agent), how should he teach the class, supposing that Bob plays first? If Alice assumes Bob is an optimal agent, how hard should she work, supposing Alice plays first?
2. We discussed the minimax algorithm in class for zero-sum games. What must we change in order to make minimax work for non-zero-sum games?
3. Draw a game tree for this problem supposing that Bob goes first. Propagate values up through the tree using (the non-zero-sum variant of) minimax search.
4. As there are more students than just Alice, it makes sense for Bob to model his class as a distribution over types of students. Suppose Bob believes that 35% of his class will work hard, 40% will work, and 25% will hardly work. Draw the expectimax tree for this setting, concentrating only on Bob's reward, and compute expected node values. What is Bob's expected reward for this setting and which type of class should he teach?