CMPSCI 683 **Fall 2002**
Prof. Lesser December 16, 2002

Final Exam

120 minutes, open book. For each question, explain your answer clearly and concisely.

1. **Learning**. Choose 11 out of 13, and answer each with a short explanation. [4 pts each]

   (a) Consider the following three approaches to classification: decision trees, instance-based learning, and neural networks. Which one would you choose if there is a lot of training cases, and each training case had 40 separate attributes? How does the issue of overfitting come into play in your decision of which approach to use?

   (b) How does the depth and breadth of the backup tree relate to both the speed and effectiveness of reinforcement learning?

   (c) Why, in instance-based learning, do you use a weighted-distance function?

   (d) Why, in case-based reasoning, may the adapting of a solution of a prior case to solve a new case require significant reasoning?

   (e) Explain the basic steps in policy-iteration as applied to solving a Markov Decision Process.

   (f) For Q learning, explain why a model of the environment is not necessary.

   (g) Explain the role of the learning factor in reinforcement/Q learning. Why, when there are probabilistic transitions, do you need a more complicated formulation of the learning factor?

   (h) What are the advantages and disadvantages of using hidden nodes in a neural network?

   (i) Why is Backpropagation tricky in a neural network?

   (j) What type of validation tests can you set up to check for overfitting? Why will they expose overfitting?

   (k) What is cross-validation and why is it needed?

   (l) Draw a step-threshold perceptron that computes the boolean implication function ($a \rightarrow b$).

   (m) In the version space algorithm, do either the computation time or the final concept learned depend on the sequence in which the examples are presented?

2. **Architecture**. Choose 5 out of 6, and answer each with a short explanation. [4 pts each]

   (a) One approach to building a distributed sensor network, where the goal is to construct an accurate view global view of the environment, was, for each agent, to only transmit its high-level partial results (based on processing its local sensor data) to other agents. Other agents would integrate this information into their own conclusions, and then send out their revised conclusions. What are the potential advantages and disadvantages of such an approach? (hint: what aspects of "satisficing" are involved in this approach?)

   (b) What are the main differences between a behavior-based architecture, such as the subsumption architecture, and a strictly hierarchical approach to an agent architecture? What would be situations where you would prefer one approach over another?

   (c) Why is being able to control the type and form of preprocessing critical to how the BB1/Guardian architecture handles real-time issues? Explain with an example of how preprocessing can be varied based on a particular situation.

   (d) When would you want to use an anytime approach versus a design-to-time approach? What are the relative advantages of each approach?

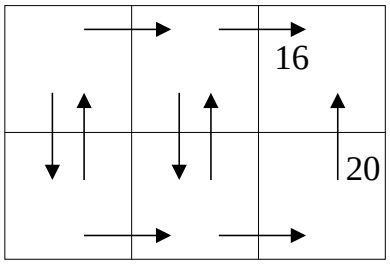
   (e) How does the Sussman Anomaly explain why a "divide and conquer" approach to planning won't always work? Motivate the need for more complicated approaches such as partial-order planning.

   (f) Explain the essential difference between a Markov policy and a plan.

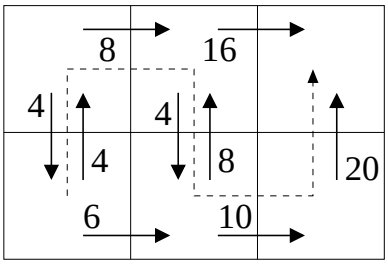
3. **Reinforcement Learning** [18 pts]

   Consider the deterministic world below (part (a)). Allowable moves are shown by arrows, and the numbers indicate the reward for performing each action. If there is no number, the reward is zero.

   Given the Q values in (b), show the changes in the Q estimates when the agent take the path shown by the dotted line (the agent starts in the lower left cell) when $\gamma = 0.5$. Show all of your work.



(a)



(b)

4. **Inductive Learning** [18 pts]

Consider the following set of examples. Each example has six attributes and one target output.

	Attribute						
	A_1	A_2	A_3	A_4	A_5	A_6	T
E_1	1	0	1	0	0	0	1
E_2	1	0	1	1	0	0	1
E_3	1	0	1	0	1	0	1
E_4	1	1	0	0	1	1	1
E_5	1	1	1	1	0	0	1
E_6	1	0	0	0	1	1	1
E_7	1	0	0	0	1	0	0
E_8	0	1	1	1	0	1	1
E_9	0	1	1	0	1	1	0
E_{10}	0	0	0	1	1	0	0
E_{11}	0	1	0	1	0	1	0
E_{12}	0	0	0	1	0	1	0
E_{13}	0	1	1	0	1	1	0
E_{14}	0	1	1	1	0	0	0

Run the decision tree learning rule, and draw the first five splits that you generate. Clearly motivate your decisions.