Massachusetts Institute of Technology

16.499 Intelligent Embedded Systems

Assignment #1 **Due: in class 4/03/00**

Background for Problems 1 and 2

Problems 1 and 2 test your understanding of Markov decision processes (MDPs) and reinforcement learning (RL). The problems are based on the lectures by Leslie Kaebling and the survey articles:

- L. Kaebling, M. Littman and A. Moore, Reinforcement learning: a survey, *Journal of Artificial Intelligence Research* , **4** (1996) 237-285.

- L. Kaebling, M. Littman and A. Cassandra, Planning and Acting in Partially Observable Stochastic Domains, *Elsevier* (1998) 237-285.

These two problems involve simple coding, which you can do in a language of your choice, such as Matlab, Scheme, Lisp, C, C++, Java …

In case you have difficulties on this problem, Leslie Kaebling has kindly agreed to answer questions by email at lpk@ai.mit.edu, as well as myself.

Consider an MDP with 5 states and 2 actions (a1, a2). Action a1 is described by the state-transition probability matrix P(s' | s, a1), which is the probability of transitioning to s' in the next state, given that you are currently in state s and perform action a1:

s \ s’	**1**	**2**	**3**	**4**	**5**
1	.1	.6	.2	0	.1
2	0	.1	0	.9	0
3	.2	.8	0	0	0
4	.1	0	.1	.1	.7
5	.8	.1	0	0	.1

a2 is described by the matrix P(s' | s, a2):

s \ s'	**1**	**2**	**3**	**4**	**5**
1	.1	.1	.1	.1	.6
2	.7	0	0	.3	0
3	.1	.8	.05	0	.05
4	.2	0	.4	.2	0
5	0	.1	0	.6	.3

The reward function R (depending only on state s in this case) is described by:

s	**R**
1	+1
2	0
3	0
4	0
5	-1

The discount factor is 0.9.

Problem 1: Markov Decision Processes

Part a. Implement the value iteration algorithm in the language of your choice (e.g., Matlab, Scheme, Lisp, C, C++, Java …). Show your commented listing.

Part b. Run value iteration on this problem with epsilon = 0.01. What is the optimal value function, V*? What is the optimal Q*(s,a) function for this value function? Finally, what is the optimal policy?

Part c. Again using epsilon = 0.01, plot |V_t - V*| as a function of iteration t. (V_t is the approximate value function computed at each iteration. |V1 - V2| is the maximum norm; that is the absolute value of the largest difference in V over the state space).

Problem 2: Reinforcement Learning

Part a. Implement the Q-learning algorithm in the language of your choice. Show your commented listing.

Part b. Implement a simulator of the MDP described above. That is, implement a procedure that, when given state s and action a, returns the appropriate reward r, and a new state s', drawn according to the distribution P(s'|s,a). Show your commented listing.

Part C. Implement an exploration strategy that picks the apparent best action with probability 0.9 and a random action with probability 0.1. Couple this to your simulator, and to your Q-learning algorithm, using a constant alpha = 0.1. Show your commented listing.

Now show a run that demonstrates that this strategy converges to the values you got in Problem 1 above.

Part D. Do the following experiment:

- Run the Q-learning algorithm for 10 steps starting in state 1.
- Freeze the Q-learning algorithm and run the policy that takes the action with the highest estimated Q-value in each state for 25 steps, starting in state 1.
- Calculate the total discounted reward received over those 25 states.

Repeat this experiment, while first running the Q-learning algorithm for 20 steps, 30 steps, etc.

Now, plot the discounted reward as a function of # of learning steps (the x-values should be 10, 20, 30, etc). This should converge to V*(1).

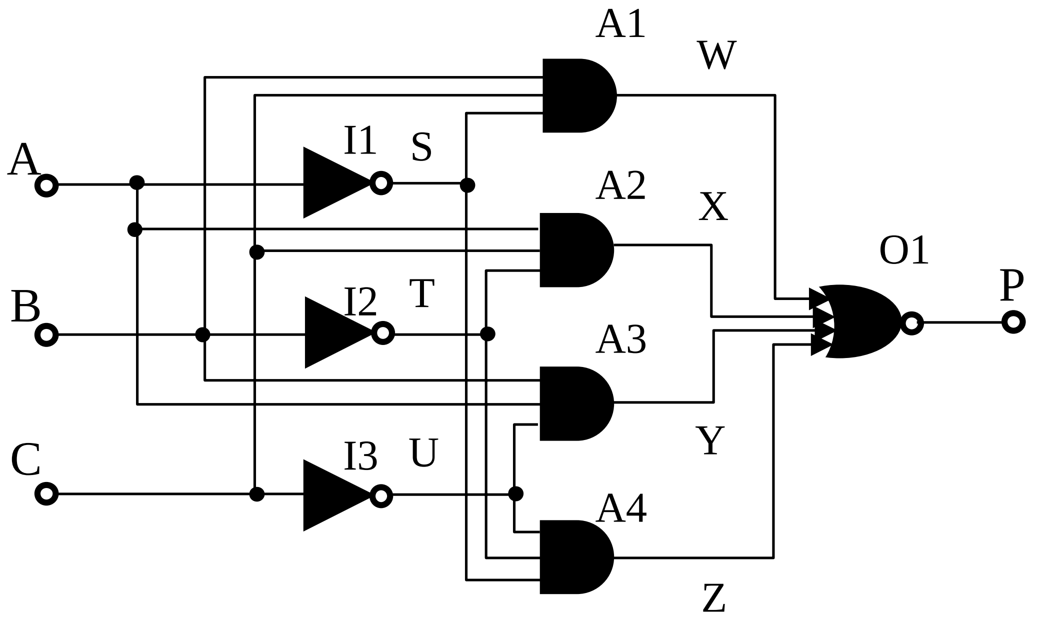
Background for Problems 3 and 4

Problems 3 and 4 test your understanding of model-based diagnosis. The problems are based on the lecture by Brian Williams and two of the articles on model-based diagnosis that were distributed in class:

- R. Davis and W. C. Hamscher, Model-based reasoning: troubleshooting , in H. E. Shrobe (Ed.) *Exploring Artificial Intelligence: Survey Talks from the National Conferences on Artificial Intelligence,* 297-346, Morgan Kaufmann, San Mateo, CA, 1988.
- J. de Kleer and B. Williams, Diagnosing multiple faults, *Artificial Intelligence* **32** (1): 97-130, April 1987.

Consider the simple three bit parity circuit shown below. This circuit has three binary inputs A, B and C, and one binary output P. P is 1 if an odd number of inputs are 1; otherwise P is 0. For example, If A =1, B = 0 and C = 1, then P =0. If we change B = 0 to B=1, then P becomes P = 1.

The circuit is composed of three inverters, I1 -- I3, feeding four and gates, A1 -- A4. The and gate outputs feed into one nor gate, O1, whose output is P. Whe working correctly, an inverter produces an output of 1 if its input is 0, and an output of 0 if its input is 1. When working correctly an And gate outputs a 1 if all its inputs are 1, otherwise it outputs 0. When working correctly a Nor gate outputs a 0 if at least one of its inputs is 1, otherwise it outputs a 1.



Problem 3: Diagnosing Single Faults

This problem tests your understanding of single fault diagnosis, first using exhaustive fault models and then using constraint suspension. Both approaches are presented in:

- R. Davis and W. C. Hamscher, Model-based reasoning: troubleshooting , in H. E. Shrobe (Ed.) *Exploring Artificial Intelligence: Survey Talks from the National Conferences on Artificial Intelligence*, 297-346, Morgan Kaufmann, San Mateo, CA, 1988.

For the three bit parity circuit you observe that the inputs are A = 1, B = 0 and C = 0, and the output is P = 0.

Part A: Diagnosis with Exhaustive Fault Modes.

First diagnose the circuit by assuming a single point of failure, and by assuming that you know an exhaustive set of fault modes for each component. The inverters (I1 -- I3) and Nor gate (O1) have a single failure mode, which is "stuck-at-1", while And gates (A1-- A4) have a single failure mode, which is "stuck-at-0." If a component is stuck-at-1, then its output is 1 independent of its inputs. Likewise, if a component is stuck-at-0, then its output is 0 independent of its inputs.

List all single fault component/failure-mode pairs that are consistent with the observations (A = 1, B = 0, C = 0, P = 0) and the component models. List all components that are exonerated from failure.

Next assume that inverters fail stuck-at-0 instead of stuck-at-1. Once again list all single fault pairs and all exonerated components.

Part B: Constraint suspension.

Assuming all components are working correctly, perform a forward prediction to detect a symptom, and then trace backwards to generate the set of "suspects". List the set of suspects and explain why you concluded they are suspects.

Next, test each suspect by suspending its constraint. Which suspects are consistent with the observations and which ones are ruled out? Why did you get this result?

Problem 4: Diagnosing Multiple Faults

This problem tests your understanding of multiple fault diagnosis using the GDE algorithm presented in:

- J. de Kleer and B. Williams, Diagnosing multiple faults, *Artificial Intelligence* **32** (1): 97-130, April 1987.

For the three bit parity circuit assume we observe A =1, B = 0, C = 1 and P = 1.

Part A: Environments

List all minimal environments for every variable (i.e., A, B, C, S, T, U, W, X, Y, Z and P).

Part B: Conflicts

List all minimal conflicts, given this set of environments.

Part C: Candidates

List all minimal candidates for the set of conflicts.

Part D: Probing

Assume the above observations, environments, conflicts and diagnoses. In addition, assume that all failures are equally likely and that a component working correctly is roughly an order of magnitude likely than it failing (i.e. P(OK) = .9 and P(Unknown) = .1).

Where should you make the next observation to learn the most about the cause of failure? Provide a commonsense explanation for your answer based on the minimum entropy equations in the second half of the de Kleer and Williams paper.

Problem 5: Path Planning Using Lazy PRM

This problem tests your understanding of the Lazy probabilistic roadmap planner (lazy PRM), and your ability to communicate the algorithm in a simple intuitive manner. Start by reading the paper:

- R. Bohlin and L. Kavraki, Path planning using lazy PRM, to appear in ICRA 2000.

This paper was distributed in class, and is available on the course web pages: www.ai.mit.edu/courses/16.499.

This problem is very non-standard, but tests your ability to flesh out and then to communicate the key ideas behind a research paper to a research team.

Suppose you are part of a team building a softball size satellite, the "Portable Satellite Assistant (PSA)". PSA navigates around the corridors of the international space station, looking for leaks in the hull or problems with a set of environmental sensors. You've been asked to give a presentation of Lazy PRM to the team as a candidate algorithm for PSA path planning.

Please assemble a set of slides that walk through the steps of PLR applied to the movement of PSA between two segments of space station. These slides correspond to the steps outlined in each subsection of section 3 of the Bohlin and Kavraki paper.

The slides should communicate visually whenever possible. Along with each slide include a text narrative (called a facing or notes page), describing that step in the algorithm.

In your presentation make sure you do three things:

- Flesh out any parts of the algorithm that are left open in the paper. For example, describe the A* algorithm mentioned in section 3.2. Select and describe one of the collision detection algorithms described in section 3.3. And so on for the remaining sections….
- Make design commitments for the lazy PLR algorithm, as suitable for the task. For example, demonstrate the algorithm on a drawing of part of space station, perhaps finding the relevant information about space station on the web (if available). In section 3.1 select a reasonable value for M_{neighb}, ρ_{coll}etc. and argue why these commitments are reasonable. Make similar commitments for each relevant subpart of section 3 of the paper.
- Finally comment on the appropriateness of Lazy PLR to this task, highlighting any features of the algorithms, concerns and areas for future exploration.

Use your judgement about length. You probably need no more than 20 slides, and 10 may prove quite sufficient.