Introduction to AI (Spr2018) Midterm Exam 4/26/2018 Solutions

1. [25%] Short questions. Be concise and clear in your answers.

(a) [3%] What is the purpose of evaluation functions in game tree search?

To reach terminal states in game tree search is usually unfeasible. We can use an evaluation function to estimate the "values" of non-terminal states, so that the search does not need to reach actual terminal states.

(b) [3%] When we say a heuristic function is admissible, what does it mean?

An admissible heuristic function is one that never overestimates the cost to the goal.

(c) [2%] Consider the bias-variance dilemma in supervised learning. Explain the meaning of bias in a regression problem such as polynomial fitting.

For a regression problem, the bias represents the difference between the target and estimated outputs for the training data.

(d) [2%] Following (c), how can you define bias for a classification problem?

For a classification problem, the bias represents the classification error rate for the training data.

(e) [2%] Following (c), if we use more terms in the polynomial, does the bias increase of decrease in general? How about the variance?

Bias will decrease (better fitting) and variance will increase.

(f) [3%] What is the purpose of "chance nodes" in the game trees of stochastic games?

A chance node is used in a game tree to handle the several possible outcomes of a chance event (such as a dice roll); with each possible outcome being a child of the chance node.

(g) [3%] What is the "degree heuristic" when solving constraint satisfaction problems with backtracking search?

When selecting a variable to assign a value, select the one with the most constraints on the remaining (unassigned) variables.

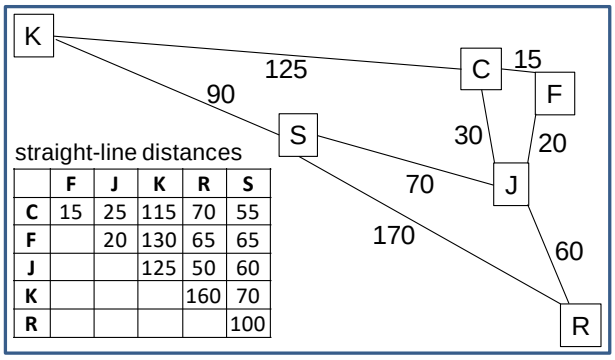
(h) [4%] Briefly explain the difference between General AI and Narrow AI. Give an example each.

The objective of General AI is to create systems/agents that exhibit human-like intelligent reasoning/learning/behaviors. Narrow AI, on the other hand, focuses on specialized systems/agents that can do specific types of tasks intelligently. Example: Many robots in science fictions, as well as some designed to look and behave humanly, are examples of General AI. Examples of Narrow AI include systems such as speech recognition or self-driving cars.

(i) [3%] What is the difference between "discrete" and "continuous" environments? Give an example each.

This generally means whether the states/actions of an agent are defined on discrete or continuous domains. For example, a chess board is a discrete environment, and self-driving cars operate in a continuous environment that is the road system.

2. [25%] We want to find paths between cities using the map to the right. Always use tree search (no checking for repeated states) when answering the problems below. For equally preferred nodes, generate and expand them in alphabetical order.



straight-line distances

	F	J	K	R	S
C	15	25	115	70	55
F		20	130	65	65
J			125	50	60
K				160	70
R					100

(a) Draw the search tree (start state: K; goal state: R) using breadth-first search. List all the expanded nodes in the correct order, and indicate the solution path found.

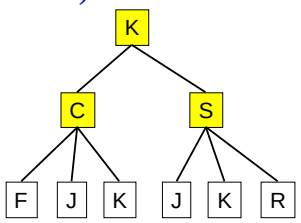
(b) Draw the A* search tree (start state: K; goal state: R) with the straight-line distance as the heuristic. Show the *h*, *g*, and *f* values of each node. List all the expanded nodes in the correct order, and indicate the solution path found.

(c) For going from R to K, list all the nodes expanded by uniform-cost search in the correct order. Also give the path found.

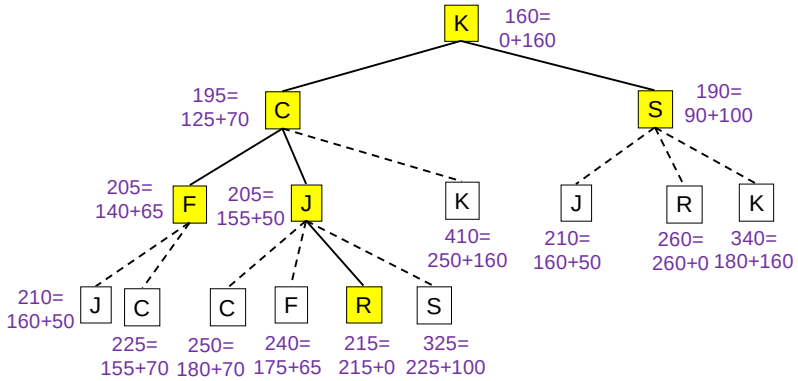
(d) Is iterative-deepening search (IDS) optimal for this type of problem (shortest-distance path)? Explain briefly.

(e) What is the problem of using iterative-deepening A* search (IDA*) for this type of problem (shortest-distance path)? Explain briefly.

(a) path: K→S→R
(Shaded nodes are the expanded nodes.)



(b) path: K→C→J→R
(Dashed lines indicate nodes that do not need to be generated because they have equal or higher costs than already generated/expanded nodes of the same state, or are later replaced by other, lower-cost nodes to the same state)
(Shaded nodes are the expanded nodes.)



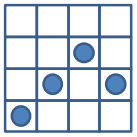
虛線的 nodes 若沒畫不扣分

(c) R J F C S K; path found: R→J→C→K

(d) IDS does not take path costs (distances) into consideration. It will find the path with the fewest number of steps, which may not be the shortest-distance path. So it is not optimal.

(e) For such a problem where the distance is a continuous quantity, IDA* basically just checks one more node for each new "deepening" iteration. This will lead to huge computational cost. (On the other hand, problems with discrete costs, such as 8-puzzle, will not have this problem.)

3. [25%] For the 4-queen problem, let P1 to P4 represent the positions of the four queens in the four columns from left to right. Example: For the configuration to the right, we have P1=4, P2=3, P3=2, P4=3. The objective of this problem is for you to apply the AC-3 procedure after the first assignment. The initial domains are all {1,2,3,4}.



(a) Assume that we assign P1=3 initially. Update the domains of P2~P4 using AC-3.

(b) Assume that we assign P1=1 initially. Update the domains of P2~P4 using AC-3.

(c) Using your results in (a) and (b) as examples, explain the possible benefits of this step (maintaining arc consistency) in backtracking search for constraint satisfaction problems.

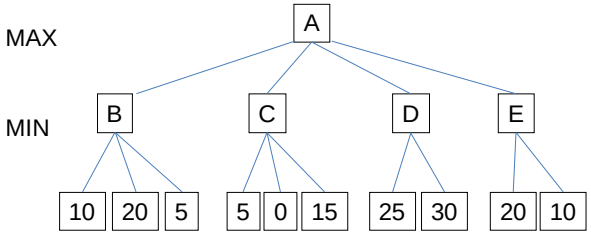
(a) Initially, domain(P1) = {3} and domain(P2) = domain(P3) = domain(P4) = {1,2,3,4} before AC-3.
Check arc P2→P1 ➔ domain(P2)→{1} ➔ changed
Check arc P3→P1 ➔ domain(P3)→{2,4} ➔ changed
Check arc P4→P1 ➔ domain(P4)→{1,2,4} ➔ changed
Check arc P1→P2 ➔ domain(P1)→{3} ➔ unchanged
Check arc P3→P2 ➔ domain(P3)→{4} ➔ changed
Check arc P4→P2 ➔ domain(P4)→{2,4} ➔ changed
Check arc P1→P3 ➔ domain(P1)→{3} ➔ unchanged
Check arc P2→P3 ➔ domain(P2)→{1} ➔ unchanged
Check arc P4→P3 ➔ domain(P4)→{2} ➔ unchanged
Check arc P1→P4 ➔ domain(P1)→{3} ➔ unchanged
Check arc P2→P4 ➔ domain(P2)→{1} ➔ unchanged
Check arc P3→P4 ➔ domain(P3)→{4} ➔ unchanged
順序不一定，但結果應相同

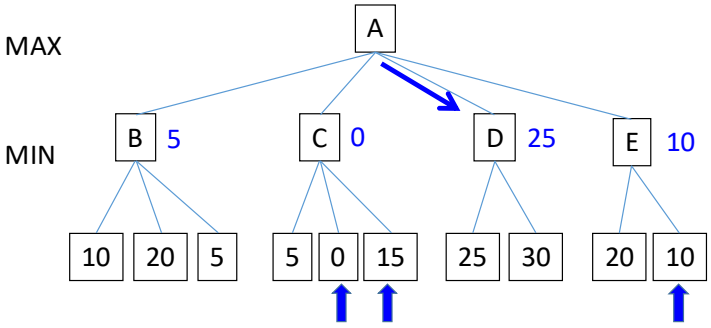
(b) Initially, domain(P1) = {1} and domain(P2) = domain(P3) = domain(P4) = {1,2,3,4} before AC-3.
Check arc P2→P1 ➔ domain(P2)→{3,4} ➔ changed

Check arc P3→P1 ➔ domain(P3)→{2,4} ➔ changed
Check arc P4→P1 ➔ domain(P4)→{2,3} ➔ changed
Check arc P1→P2 ➔ domain(P1)→{1} ➔ unchanged
Check arc P3→P2 ➔ domain(P3)→{2} ➔ changed
Check arc P4→P2 ➔ domain(P4)→{2,3} ➔ unchanged
Check arc P1→P3 ➔ domain(P1)→{1} ➔ unchanged
Check arc P2→P3 ➔ domain(P2)→{4} ➔ changed
Check arc P4→P3 ➔ domain(P4)→empty ➔ failure (AC-3 can terminate here)
順序不一定，但結果應相同

(c) Considering the use of AC-3 after each assignment in backtracking search:
AC-3 can narrow down the domains of the variables. This will allow the backtracking search to focus on variable values that are unlikely to violate arc consistencies.
In some cases (as in (a)), AC-3 is able to narrow down all the domains down to just one value, resulting in a solution without further search. Therefore, the backtracking search can be terminated with the solution returned.
In some cases (as in (b)), AC-3 results in some domains becoming empty. This indicates that the assignment just made (e.g., the "P1=1" assignment in (b)) will lead to no solution, meaning that the search should backtrack. This reduces to computational cost of growing a part of the search tree that has no solution.

4. [10%] Given the game tree to the right:
(a) Give the minimax values for nodes B-E. For minimax search, which action should MAX take?
(b) Which nodes are skipped by α–β pruning? You can just mark them on the figure.





5. [15%] Your task is to learn a decision tree for a two-class classification problem. There are 8 training samples and 4 attributes, which are given in the table. Use maximum information gain (minimum remaining entropy) to choose the attributes to split the nodes. The formula of entropy is $H(V) = -\sum_k P(v_k)\log_2 P(v_k)$, where v_k is the kth possible value of the random variable V.

Samples	Attributes				Output
	SL	SW	PL	PW	Class
$\boldsymbol{x}_1$	1	1	1	1	A
$\boldsymbol{x}_2$	2	2	1	1	A
$\boldsymbol{x}_3$	1	2	1	1	A
$\boldsymbol{x}_4$	2	1	1	1	A
$\boldsymbol{x}_5$	1	1	1	2	B
$\boldsymbol{x}_6$	2	1	2	1	B
$\boldsymbol{x}_7$	2	2	2	1	B
$\boldsymbol{x}_8$	2	2	2	2	B

Numbers that you can use: Let $f(p) = -p\log_2 p$

$f(0)=0$, $f(1)=0$, $f(1/2)=0.5$, $f(1/3)\approx 0.53$,
$f(1/4)=0.5$, $f(1/5)\approx 0.46$, $f(2/3)\approx 0.39$,
$f(2/5)\approx 0.53$, $f(3/5)\approx 0.44$, $f(4/5)\approx 0.26$

To select the first attribute, we compute the remaining entropies when the samples are split using each of the attributes.

Split by SL: 3 samples (2A,1B) for SL=1, 5 samples (2A,3B) for SL=2

Remainder(SL):

$$\frac{3}{8}[f(2/3)+f(1/3)]+\frac{5}{8}[f(2/5)+f(3/5)]\approx\frac{1}{8}[3(0.39+0.53)+5(0.39+0.53)]=\frac{7.36}{8}$$

Split by SW: 4 samples (2A,2B) for SW=1, 4 samples (2A,2B) for SW=2

Remainder(SW): $\frac{4}{8}[f(2/4)+f(2/4)]+\frac{4}{8}[f(2/4)+f(2/4)]=\frac{1}{8}[4(0.5+0.5)+4(0.5+0.5)]=\frac{8}{8}$

Split by PL: 5 samples (4A,1B) for PL=1, 3 samples (3B) for PL=2

Remainder(PL): $\frac{5}{8}[f(4/5)+f(1/5)]+\frac{3}{8}[f(3/3)+f(0/3)]\approx\frac{1}{8}[5(0.26+0.46)+3(0+0)]=\frac{3.6}{8}$

Split by PW: 6 samples (4A,2B) for PW=1, 2 samples (2B) for PW=2

Remainder(PW): $\frac{6}{8}[f(4/6)+f(2/6)]+\frac{2}{8}[f(0/2)+f(2/2)]\approx\frac{1}{8}[6(0.39+0.53)+2(0+0)]=\frac{5.52}{8}$

Remainder(PL) is the smallest. So the first attribute is PL.

Now we need to choose the second attribute.

For the subtree of PL=1:

Split by SL: 3 samples (2A,1B) for SL=1, 2 samples (2B) for SL=2

Remainder(SL): $\frac{3}{5}[f(2/3)+f(1/3)]+\frac{2}{5}[f(0/2)+f(2/2)]\approx\frac{1}{5}[3(0.39+0.53)+2(0+0)]=\frac{2.76}{5}$

Split by SW: 3 samples (2A,1B) for SW=1, 2 samples (2B) for SW=2

Remainder(SW): $\frac{3}{5}[f(2/3)+f(1/3)]+\frac{2}{5}[f(0/2)+f(2/2)]\approx\frac{1}{5}[3(0.39+0.53)+2(0+0)]=\frac{2.76}{5}$

Split by PW: 4 samples (4A) for PW=1, 1 sample (1B) for PW=2

Remainder(PW) = 0

So we split by PW. The subtree terminates here as there is no need to go any further.

For the subtree of PL=2:

All 3 samples are class B, so there is no need to go any further.

The resulting tree is shown to the right.