

**Course title: Introduction to Artificial Intelligence**

Course code: **MAT1206E**

Credits: **3**

Exam No.: **01**

For students of course classes: **MAT1206E 1, MAT1206E 2**

Time allowed: **90 minutes** (not including the time for handing out the paper)

**Question 1.** (1 point) A spam filter is an agent that puts each incoming email into one of two classes, *wanted* or *spam*, and deletes the emails classified as spam. Two spam filters were tested on the same 1000 emails with the following results.

| Agent 1     |        | correct class |      |
|-------------|--------|---------------|------|
|             |        | wanted        | spam |
| spam filter | wanted | 196           | 30   |
| decides     | spam   | 4             | 770  |

| Agent 2     |        | correct class |      |
|-------------|--------|---------------|------|
|             |        | wanted        | spam |
| spam filter | wanted | 200           | 50   |
| decides     | spam   | 0             | 750  |

- (a) State the goal of a cost-based agent.
- (b) A spam email that is not filtered out costs the user 1 cent, because it has to be deleted manually. A wanted email that is deleted costs the user 100 cents. Compute the total cost caused by the errors of each agent and decide which agent is better. Explain why comparing only the numbers of errors is not sufficient here.

**Question 2.** (2.5 points)

- (a) A delivery robot works with the following knowledge base  $KB$ : (1) If the battery is low, then the robot returns to the charging station or it sends a warning. (2) The robot sends a warning if and only if the network is available. (3) The network is not available. (4) The battery is low. Use the propositional variables  $L$  (the battery is low),  $R$  (the robot returns to the charging station),  $W$  (the robot sends a warning) and  $N$  (the network is available).
- (i) Formalize  $KB$  and transform it into conjunctive normal form (CNF). Number the clauses.
- (ii) Prove  $KB \models R$  with the resolution calculus. Write every resolution step as  $\text{Res}(m, n)$ : (resolvent) $_k$ , where the resolvent of clauses  $m$  and  $n$  is numbered  $k$ .
- (b) Consider the following statements: (1) For every customer, there is a baker who likes this customer. (2) Everyone who is liked by at least one baker is happy. (3) Bob is a customer. Use the predicates  $customer(x)$  ( $x$  is a customer),  $baker(x)$  ( $x$  is a baker),  $likes(x, y)$  ( $x$  likes  $y$ ),  $happy(x)$  ( $x$  is happy) and the constant  $bob$  (Bob).
- (i) Formalize the statements in first-order predicate logic and transform them into conjunctive normal form (CNF) for use in the resolution proof in part (ii). Number the clauses. If a Skolem function is needed, call it  $s$ .
- Recall that resolution is applied to clauses whose variables are implicitly universally quantified. Skolemization eliminates existential quantifiers while preserving satisfiability: the original formula is satisfiable if and only if the Skolemized formula is satisfiable. However, the two formulas need not be logically equivalent.
- (ii) Prove by resolution that Bob is happy. For every resolution step, give the two parent clauses, the most general unifier (MGU) and the resolvent.

**Question 3.** (0.5 points) A student claims that first-order predicate logic (PL1) is decidable, with the following argument: “We take a complete proof calculus for PL1. With it, we can find a proof for any true formula in finite time. For every other formula  $\phi$ , I apply the calculus to  $\neg\phi$  and show that  $\neg\phi$  is true. Thus  $\phi$  is false. Thus I can prove or refute every formula in PL1.” Find the mistake in this argument, and correct the argument: for which formulas does the corrected procedure give an answer in finite time? Why is propositional logic decidable?

**Question 4.** (2 points) Consider the following PROLOG program with numbered clauses, where `just_ate(X,Y)` means that X has just eaten Y and `is_digesting(X,Y)` that X is digesting Y.

```

1  is_digesting(X,Y) :- just_ate(X,Y).
2  is_digesting(X,Y) :- just_ate(X,Z), is_digesting(Z,Y).
3  just_ate(stork,frog).
4  just_ate(frog,fly).

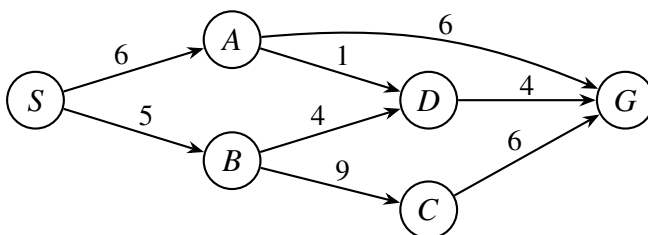
```

- (a) Which clause of `is_digesting` is the base clause and which is the recursive clause? State the declarative meaning of clause 2.
- (b) Draw the search tree for the query `?- is_digesting(stork,Y)`. Each box contains the list of remaining goals, each edge is labelled with the number of the clause tried for the selected goal (a clause of the same predicate whose head does not unify leads to a fail leaf), and each leaf is `[]` (with the answer) or `fail`. Give the answers in the order in which PROLOG returns them, and explain this order.
- (c) Clause 2 is replaced by the following clause, which has the same declarative meaning:

2' `is_digesting(X,Y) :- is_digesting(Z,Y), just_ate(X,Z).`

What happens when the query of (b) is asked again and all answers are requested? Explain why.

**Question 5.** (2 points) In the search problem below,  $S$  is the starting state,  $G$  is the only goal state, the arrows are the only actions (labelled with their costs), and the table gives the admissible heuristic cost estimate  $h$ .



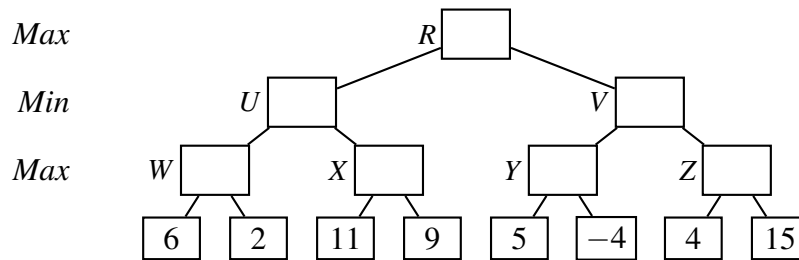
| state $s$ | $S$ | $A$ | $B$ | $C$ | $D$ | $G$ |
|-----------|-----|-----|-----|-----|-----|-----|
| $h(s)$    | 8   | 4   | 4   | 3   | 4   | 0   |

- (a) (i) Apply A\* search, that is, HEURISTICSEARCH with  $f(n) = g(n) + h(n)$  as the heuristic evaluation function, where  $g(n)$  is the cost of the path from the root to  $n$ . Write each search node as its path from  $S$ , for example  $S-A$ , and insert every generated node into `NodeList`, even if `NodeList` already contains a node with the same state. Draw and complete the table below, adding one row for each selected node. Record its  $g, h, f$  values and `NodeList` after the expansion, sorted by increasing  $f$ , with each node's  $f$ -value in parentheses. In the final row, record why the search stops in the last column.

| Step     | Selected node | $g$ | $h$ | $f$ | NodeList after the expansion ( $f$ -values) |
|----------|---------------|-----|-----|-----|---------------------------------------------|
| 1        |               |     |     |     |                                             |
| $\vdots$ |               |     |     |     |                                             |

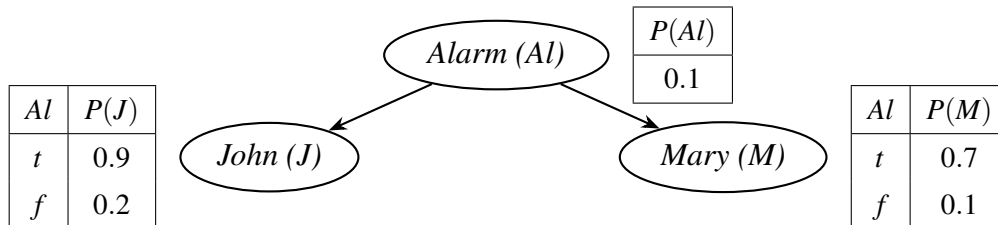
Which path does A\* return, and what is its cost?

- (ii) During the search, the node  $S-A-G$ , whose state is the goal  $G$ , is generated. Why does A\* not stop at that moment?
- (b) In the game tree below, Max moves at the root  $R$ ; the ratings of the leaves are given from Max's point of view. Compute the minimax value of every inner node. Which move does Max choose at  $R$ ?



**Question 6.** (2 points)

- (a) The Bayesian network below has the binary variables  $Al$  (the alarm sounds),  $J$  (John calls) and  $M$  (Mary calls), each with the values  $t$  and  $f$ . Throughout this question,  $P(X)$  abbreviates  $P(X = t)$  and  $\neg X$  stands for  $X = f$ ; for example, the table of  $J$  gives  $P(J | Al) = 0.9$  and  $P(J | \neg Al) = 0.2$ .



- (i) Write  $P(Al, J, M)$  as a product of probabilities given by the network.
- (ii) Compute  $P(J)$  and  $P(Al | J)$ .
- (iii) Compute  $P(M | J)$  and  $P(M)$ . Are  $J$  and  $M$  independent? Are they conditionally independent given  $Al$ ? Justify your answers.
- (b) For two binary variables  $A$  and  $B$ , only  $P(A) = 0.6$  and  $P(A \vee B) = 0.9$  are known. Let  $p_1, p_2, p_3, p_4$  denote  $P(A, B), P(A, \neg B), P(\neg A, B), P(\neg A, \neg B)$ , respectively.
- (i) Write the given knowledge and the normalization condition as equations in  $p_1, \dots, p_4$ .
- (ii) Determine the distribution  $(p_1, p_2, p_3, p_4)$  selected by the principle of maximum entropy (MaxEnt). Justify the values that are not fixed by the equations.
- (iii) Compute  $P(B)$  and  $P(B | A)$  for this distribution.

VIETNAM NATIONAL UNIVERSITY, HANOI  
VNU UNIVERSITY OF SCIENCE

ANSWER KEY AND MARKING SCHEME  
MIDTERM EXAMINATION, SEMESTER I, ACADEMIC YEAR 2026–2027  
Course title: Introduction to Artificial Intelligence

Course code: **MAT1206E** Credits: **3** Exam No.: **01**  
For students of course classes: **MAT1206E 1, MAT1206E 2**

**Solution 1.** [1 point]

|                                                                                                                                                                                                                                                                                    |             |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| (a) A cost-based agent tries to minimize the long-term cost, that is, the average cost, caused by its wrong decisions. The total cost is the sum of all errors, each weighted with its cost factor.                                                                                | <b>0.25</b> |
| (b) Agent 1 deletes 4 wanted emails and lets 30 spam emails through: $4 + 30 = 34$ errors with cost $4 \cdot 100 + 30 \cdot 1 = 430$ cents. Agent 2 deletes no wanted email and lets 50 spam emails through: $0 + 50 = 50$ errors with cost $0 \cdot 100 + 50 \cdot 1 = 50$ cents. | <b>0.5</b>  |
| Agent 2 is better, because its cost is much lower, although it makes more errors ( $50 > 34$ ). The two kinds of errors have very different severity (100 cents versus 1 cent), so each error must be weighted with its cost; the number of errors ignores this difference.        | <b>0.25</b> |

**Solution 2.** [2.5 points]

|                                                                                                                                                                                                                                                                                                                                                                            |             |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| (a)(i) Formalization: $KB \equiv (L \Rightarrow R \vee W) \wedge (W \Leftrightarrow N) \wedge \neg N \wedge L$ .                                                                                                                                                                                                                                                           | <b>0.25</b> |
| Eliminating the implication and the equivalence gives $L \Rightarrow R \vee W \equiv \neg L \vee R \vee W$ and $W \Leftrightarrow N \equiv (W \Rightarrow N) \wedge (N \Rightarrow W) \equiv (\neg W \vee N) \wedge (\neg N \vee W)$ . Hence<br>$KB \equiv (\neg L \vee R \vee W)_1 \wedge (\neg W \vee N)_2 \wedge (\neg N \vee W)_3 \wedge (\neg N)_4 \wedge (L)_5.$     | <b>0.25</b> |
| (a)(ii) Add the negated query $(\neg R)_6$ to $KB$ . One possible proof: Res(1,5): $(R \vee W)_7$ , Res(6,7): $(W)_8$ , Res(2,8): $(N)_9$ , Res(4,9): $(\ )_{10}$ . The empty clause shows that $KB \wedge \neg R$ is unsatisfiable; therefore $KB \models R$ .                                                                                                            | <b>0.5</b>  |
| (b)(i) Formalization of the three statements:<br>$\forall y (customer(y) \Rightarrow \exists x (baker(x) \wedge likes(x, y))), \quad \forall x \forall y (baker(x) \wedge likes(x, y) \Rightarrow happy(y)),$ $customer(bob).$<br>The equivalent form $\forall y (\exists x (baker(x) \wedge likes(x, y)) \Rightarrow happy(y))$ of the second statement is also accepted. | <b>0.25</b> |

| <p>CNF of the first statement: eliminating the implication and moving <math>\exists x</math> to the front gives the prenex normal form <math>\forall y \exists x (\neg customer(y) \vee (baker(x) \wedge likes(x, y)))</math>, and the distributive law gives</p> $\forall y \exists x ((\neg customer(y) \vee baker(x)) \wedge (\neg customer(y) \vee likes(x, y))).$ <p>The variable <math>x</math> depends on <math>y</math>, so Skolemization replaces <math>x</math> by <math>s(y)</math>; then the universal quantifier is deleted. With the other two statements, the clauses are</p> $(\neg customer(y) \vee baker(s(y)))_1, \quad (\neg customer(y) \vee likes(s(y), y))_2,$ $(\neg baker(x) \vee \neg likes(x, y) \vee happy(y))_3, \quad (customer(bob))_4.$                                                                                                                                                                             | <b>0.5</b>                              |                                                       |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|-------------------------------------------------------|-----------|----|-----------------------------------------|--------------------------------|----|----------------------|--------------------------------|----|----------------------|-----------------------------------|----|-----------------|--------------------------------|-----|------|-------------------------------------------------------|-------------|
| <p>(b)(ii) The query is <math>Q \equiv happy(bob)</math>; add the negated query <math>(\neg happy(bob))_5</math>. One possible proof (the parent clauses never share variables):</p> <table><tr><th>Step</th><th>Resolvent</th><th>Proved by</th></tr><tr><td>6.</td><td><math>\neg baker(x) \vee \neg likes(x, bob)</math></td><td>Res(3, 5), MGU <math>\sigma: y/bob</math></td></tr><tr><td>7.</td><td><math>likes(s(bob), bob)</math></td><td>Res(2, 4), MGU <math>\sigma: y/bob</math></td></tr><tr><td>8.</td><td><math>\neg baker(s(bob))</math></td><td>Res(6, 7), MGU <math>\sigma: x/s(bob)</math></td></tr><tr><td>9.</td><td><math>baker(s(bob))</math></td><td>Res(1, 4), MGU <math>\sigma: y/bob</math></td></tr><tr><td>10.</td><td><math>()</math></td><td>Res(8, 9), MGU <math>\varepsilon</math> (the empty substitution)</td></tr></table> <p>The empty clause is derived, so <math>happy(bob)</math> follows: Bob is happy.</p> | Step                                    | Resolvent                                             | Proved by | 6. | $\neg baker(x) \vee \neg likes(x, bob)$ | Res(3, 5), MGU $\sigma: y/bob$ | 7. | $likes(s(bob), bob)$ | Res(2, 4), MGU $\sigma: y/bob$ | 8. | $\neg baker(s(bob))$ | Res(6, 7), MGU $\sigma: x/s(bob)$ | 9. | $baker(s(bob))$ | Res(1, 4), MGU $\sigma: y/bob$ | 10. | $()$ | Res(8, 9), MGU $\varepsilon$ (the empty substitution) | <b>0.75</b> |
| Step                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | Resolvent                               | Proved by                                             |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
| 6.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | $\neg baker(x) \vee \neg likes(x, bob)$ | Res(3, 5), MGU $\sigma: y/bob$                        |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
| 7.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | $likes(s(bob), bob)$                    | Res(2, 4), MGU $\sigma: y/bob$                        |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
| 8.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | $\neg baker(s(bob))$                    | Res(6, 7), MGU $\sigma: x/s(bob)$                     |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
| 9.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | $baker(s(bob))$                         | Res(1, 4), MGU $\sigma: y/bob$                        |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |
| 10.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | $()$                                    | Res(8, 9), MGU $\varepsilon$ (the empty substitution) |           |    |                                         |                                |    |                      |                                |    |                      |                                   |    |                 |                                |     |      |                                                       |             |

**Solution 3.**

[0.5 points]

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| <p>The mistake: a formula that is not true (valid) need not have a true negation. If <math>\phi</math> is satisfiable but not valid, then both <math>\phi</math> and <math>\neg\phi</math> have a model, so neither <math>\phi</math> nor <math>\neg\phi</math> is valid; the calculus finds a proof of neither, and the search may never terminate. Naming the other gap is also accepted: the procedure cannot decide in finite time whether <math>\phi</math> is true, because the search for a proof of a formula that is not true may never stop. Corrected argument: apply the calculus to <math>\phi</math> and to <math>\neg\phi</math> alternately; then every true formula is proved, and every unsatisfiable formula <math>\phi</math> is refuted by a proof of <math>\neg\phi</math>, in finite time, while for satisfiable formulas that are not valid the procedure may not terminate. An answer that distinguishes the three cases without the alternation is also accepted: valid (a proof of <math>\phi</math>), unsatisfiable (a proof of <math>\neg\phi</math>), satisfiable but not valid (possibly no answer). This agrees with the theorem that the set of valid PL1 formulas is semidecidable but not decidable.</p> | <b>0.25</b> |
| <p>Propositional logic is decidable because the truth-table method checks all <math>2^n</math> truth assignments of a formula with <math>n</math> variables, and so finds all its models, in finite time.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | <b>0.25</b> |

**Solution 4.**

[2 points]

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |             |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| (a) Clause 1 is the base clause; clause 2 is the recursive clause, because its body calls <code>is_digesting</code> again. Declarative meaning of clause 2: if X has just eaten Z and Z is digesting Y, then X is digesting Y too.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | <b>0.5</b>  |
| (b) Search tree (the selected goal is blue; Z1, Z2 belong to fresh copies of clause 2; the same tree in another clear notation, also without colour, is accepted): <div data-bbox="255 504 1157 1344"> </div>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | <b>0.75</b> |
| PROLOG returns first $Y = \text{frog}$ and then $Y = \text{fly}$ . PROLOG searches depth-first from left to right and tries the clauses from top to bottom; because the base clause 1 comes before the recursive clause 2, the direct answer (stork has just eaten frog) is found before the indirect one (stork–frog–fly). After that, every remaining branch ends in fail.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | <b>0.25</b> |
| (c) PROLOG still returns $Y = \text{frog}$ (clause 1) and then $Y = \text{fly}$ , but when further answers are requested it does not terminate; it never gives the answer false. In practice, the PROLOG system stops with a stack overflow after the two answers; this answer is also accepted. The first subgoal of clause 2' is the recursive call <code>is_digesting(Z, Y)</code> with a new, unbound Z. For this goal, PROLOG first tries clause 1: the binding $Z = \text{stork}$ , $Y = \text{frog}$ fails at the remaining goal <code>just_ate(stork, stork)</code> , and the binding $Z = \text{frog}$ , $Y = \text{fly}$ gives the answer fly. On backtracking, PROLOG tries clause 2', which creates the goal <code>is_digesting(Z1, Y)</code> ; for this goal the branches through clause 1 end in fail, and clause 2' creates <code>is_digesting(Z2, Y)</code> , and so on. In the branch that always uses clause 2', the leftmost goal is always a new <code>is_digesting</code> goal, so the <code>just_ate</code> goals behind it are never selected. Thus the search tree has an infinitely deep branch, and depth-first search never finishes it. The declarative meaning is unchanged, but the procedural meaning is not: termination depends on the order of the clauses and of the subgoals. | <b>0.5</b>  |

**Solution 5.**

[2 points]

| (a)(i) A* search; a node is written as its path, and the parenthesized number is its value $f = g + h$ :                                                                                                                                                                                                                                                                                                                       |               |     |     |     |                                                          | <b>1</b>    |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|-----|-----|-----|----------------------------------------------------------|-------------|
| Step                                                                                                                                                                                                                                                                                                                                                                                                                           | Selected node | $g$ | $h$ | $f$ | NodeList after the expansion ( $f$ -values)              |             |
| 1                                                                                                                                                                                                                                                                                                                                                                                                                              | $S$           | 0   | 8   | 8   | $S-B$ (9), $S-A$ (10)                                    |             |
| 2                                                                                                                                                                                                                                                                                                                                                                                                                              | $S-B$         | 5   | 4   | 9   | $S-A$ (10), $S-B-D$ (13), $S-B-C$ (17)                   |             |
| 3                                                                                                                                                                                                                                                                                                                                                                                                                              | $S-A$         | 6   | 4   | 10  | $S-A-D$ (11), $S-A-G$ (12), $S-B-D$ (13), $S-B-C$ (17)   |             |
| 4                                                                                                                                                                                                                                                                                                                                                                                                                              | $S-A-D$       | 7   | 4   | 11  | $S-A-D-G$ (11), $S-A-G$ (12), $S-B-D$ (13), $S-B-C$ (17) |             |
| 5                                                                                                                                                                                                                                                                                                                                                                                                                              | $S-A-D-G$     | 11  | 0   | 11  | goal node selected: stop                                 |             |
| A* returns the path $S-A-D-G$ with cost $6 + 1 + 4 = 11$ .                                                                                                                                                                                                                                                                                                                                                                     |               |     |     |     |                                                          | <b>0.25</b> |
| (a)(ii) A* stops only when a goal node is <i>selected</i> , that is, when it is the first node of NodeList, not when it is generated. When $S-A-G$ is generated (step 3), its value is $f = 12$ , but $S-A-D$ has the lower value $f = 11$ . Since $h$ is admissible, $f(S-A-D) = 11$ does not overestimate the cost of a solution through $S-A-D$ , so a cheaper solution may still exist; indeed $S-A-D-G$ costs $11 < 12$ . |               |     |     |     |                                                          | <b>0.25</b> |
| (b) Minimax values: $W = \max(6, 2) = 6$ , $X = \max(11, 9) = 11$ , $U = \min(6, 11) = 6$ ; $Y = \max(5, -4) = 5$ , $Z = \max(4, 15) = 15$ , $V = \min(5, 15) = 5$ ; $R = \max(6, 5) = 6$ . Max chooses the move to $U$ , the child with the larger minimax value.                                                                                                                                                             |               |     |     |     |                                                          | <b>0.5</b>  |

**Solution 6.**

[2 points]

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |  |             |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|-------------|
| (a)(i) By the Bayesian-network factorization, $P(Al, J, M) = P(Al)P(J   Al)P(M   Al)$ . The value of this product, $0.1 \cdot 0.9 \cdot 0.7 = 0.063$ , is also accepted.                                                                                                                                                                                                                                                                                                                     |  | <b>0.25</b> |
| (a)(ii) Marginalization:<br><br>$P(J) = P(J   Al)P(Al) + P(J   \neg Al)P(\neg Al) = 0.9 \cdot 0.1 + 0.2 \cdot 0.9 = 0.09 + 0.18 = 0.27$<br>Bayes' theorem: $P(Al   J) = \frac{P(J   Al)P(Al)}{P(J)} = \frac{0.09}{0.27} = \frac{1}{3} \approx 0.333$ .                                                                                                                                                                                                                                       |  | <b>0.5</b>  |
| (a)(iii) Given its parent $Al$ , the node $M$ is conditionally independent of $J$ , which is neither a parent nor a descendant of $M$ . Conditioning on $Al$ therefore gives<br><br>$P(M   J) = P(M   Al)P(Al   J) + P(M   \neg Al)P(\neg Al   J) = 0.7 \cdot \frac{1}{3} + 0.1 \cdot \frac{2}{3} = 0.3$<br>Marginalization gives $P(M) = 0.7 \cdot 0.1 + 0.1 \cdot 0.9 = 0.16$ . Any correct method is accepted, for example $P(M   J) = \frac{P(J, M)}{P(J)} = \frac{0.081}{0.27} = 0.3$ . |  | <b>0.25</b> |
| Since $P(M   J) = 0.3 \neq 0.16 = P(M)$ , $J$ and $M$ are not independent: observing that John calls raises the probability of an alarm, and therefore the probability that Mary calls. They are conditionally independent given $Al$ : by the network semantics above,<br><br>$P(J = j, M = m   Al = a) = P(J = j   Al = a)P(M = m   Al = a)$<br>for all values $j, m, a \in \{t, f\}$ .                                                                                                    |  | <b>0.25</b> |
| (b)(i) $P(A) = p_1 + p_2 = 0.6$ ; $P(A \vee B) = p_1 + p_2 + p_3 = 0.9$ ; normalization: $p_1 + p_2 + p_3 + p_4 = 1$ . The last two equations give $p_4 = 0.1$ .                                                                                                                                                                                                                                                                                                                             |  | <b>0.25</b> |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |             |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|
| (b)(ii) The equations give $p_4 = 0.1$ , $p_3 = 1 - 0.6 - 0.1 = 0.3$ and only $p_1 + p_2 = 0.6$ ; one equation is missing. MaxEnt chooses, among all distributions satisfying the equations, the one with the largest entropy $H = -\sum_{i=1}^4 p_i \ln p_i$ , that is, the one that adds the least information. The variables $p_1$ and $p_2$ occur symmetrically in the constraints, so they are indifferent and the maximum lies at $p_1 = p_2 = 0.3$ (maximizing $-p_1 \ln p_1 - p_2 \ln p_2$ subject to $p_1 + p_2 = 0.6$ with a Lagrange multiplier gives the same result). Hence $(p_1, p_2, p_3, p_4) = (0.3, 0.3, 0.3, 0.1)$ . | <b>0.25</b> |
| (b)(iii) $P(B) = p_1 + p_3 = 0.6$ and $P(B   A) = \frac{p_1}{p_1 + p_2} = \frac{0.3}{0.6} = 0.5$ .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | <b>0.25</b> |