UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR08010 INFORMATICS 2D: REASONING AND AGENTS

Wednesday 14th August 2013

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

1. **Answer Parts A, B and C. The multiple choice questions in Part A are worth 50% in total and are each worth the same amount. Mark one answer only for each question - multiple answers will score 0. Marks will not be deducted for incorrect multiple choice exam answers. Raw multiple choice scores may be rescaled at the discretion of the exam board. Parts B and C are each worth 25%. Answer ONE question from Part B and ONE question from Part C.**
2. **Use the special mark sheet for Part A. Answer Parts B and C each in a separate script book.**

CALCULATORS ARE PERMITTED.

Convener: J Bradfield
External Examiner: A Preece

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

Part A
ANSWER ALL QUESTIONS IN PART A. Use the special mark sheet.

1. In the Situation Calculus, what is a FRAME rule?

   (a) A rule that specifies the background context in which an action is executed.

   (b) A rule that specifies which facts become false as a result of an action.

   (c) A rule that specifies which facts are unchanged by an action.

   (d) A rule that allows the problem to be framed as a first-order logic statement.

   (e) A rule that specifies which facts become true as a result of an action.

2. Consider a Wumpus World consisting of a 5x5 grid with the following propositional 'variables' associated with each square: $S_{i,j}, B_{i,j}, P_{i,j}, OK_{i,j}$. How many assignments of truth values are there for a propositional formula containing all these variables?

   (a) $4 \times 5 \times 5$.

   (b) $2^4 \times 5 \times 5$.

   (c) $4^{2\times5\times5}$.

   (d) $2^{4\times5\times5}$.

   (e) $2^{5\times5} \times 4$.

3. If x is the maximum branching factor of a search tree, y is the depth of the least cost solution and z is the maximum depth of the search space, what is the space complexity of iterative deepening search?

   (a) $O(x^z)$.

   (b) $O(y^x)$.

   (c) $O(x^y)$.

   (d) $O(xz)$.

   (e) $O(xy)$.

4. Which of the following clauses, if any, is the result of resolving $\neg P(F(x)) \vee P(F(a))$ against a copy of itself?

   (a) True.

   (b) $\neg P(F(x)) \vee P(F(a))$.

   (c) False (the empty clause).

   (d) Resolution fails.

   (e) $\neg P(F(a)) \vee P(F(a))$.

5. Which of the following is *not* part of the DPLL algorithm, as given in Russell & Norvig and the lectures?

   (a) A sentence is true if any of its clauses is true.

   (b) A clause is true if one of its literal is true.

   (c) A literal containing a pure symbol can be set to true.

   (d) A literal in a unit clause can be set to true.

   (e) A clause where all but one literal is false can be set to true.

6. Consider a CSP problem with 5 variables X_1, X_2, X_3, X_4, and X_5, each with domain $\{1, 2, 3, 4, 5\}$. If the constraints between the variables require that $X_i > X_{i+1}$ for $1 \leq i < 5$, what will the domain of X_1 be after enforcing arc-consistency of *only* the arcs $X_2 \rightarrow X_3$ *then* $X_1 \rightarrow X_2$?

   (a) $\{\}$.

   (b) $\{1, 2, 3, 4, 5\}$.

   (c) $\{3, 4, 5\}$.

   (d) $\{1, 2, 3\}$.

   (e) $\{5\}$.

7. Which of the following statements is *true* of forward-chaining?

   (a) It always terminates for definite clauses with no function symbols.

   (b) It never terminates unless stopped by the user.

   (c) It is unsound when the search space contains loops.

   (d) It always terminates for clauses with function symbols.

   (e) It is widely used for logic programming.

8. Which of the following propositional logic formulae, if any, is a tautology?

   (a) $P \wedge \neg P$.

   (b) $(P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow Q))$.

   (c) $(Q \wedge (P \rightarrow Q)) \rightarrow P$.

   (d) $\neg P \rightarrow (\neg P \rightarrow Q)$.

   (e) None of the above.

9. Which of the following sets of clauses is the clausal normal form (CNF) of the formula $\forall x.(P(x) \leftrightarrow \exists y.Q(x, y))$?

   (a) $\{\neg P(x) \vee Q(x, F(x)), \neg Q(x, F(x)) \vee P(x)\}$

   (b) $\{\neg P(x) \vee Q(x, y), \neg Q(x, y) \vee P(x)\}$

   (c) $\{\neg P(x) \vee Q(x, F(x)), \neg Q(x, y) \vee P(x)\}$

   (d) $\{\neg P(x) \vee Q(x, F(x)), \neg Q(x, F(x)) \vee P(x)\}$

   (e) $\{\neg P(x) \vee Q(x, F(x)) \vee \neg Q(x, y) \vee P(x)\}$

10. Which of the following statements about WALKSAT (as given in Russell & Norvig and the lectures) is *false?*

    (a) It is a local search algorithm.

    (b) It uses a heuristic which minimizes the number of unsatisfied clauses.

    (c) It is a random-walk algorithm.

    (d) It is not a decision procedure for propositional logic.

    (e) It is usually much slower than DPLL for satisfiable problems.

11. Which of the following is *not* a weakness of search-based problem solving methods compared to planning?

    (a) The lack of goal-directedness in considering which actions to perform.

    (b) The inability to decompose a complex goal into subgoals.

    (c) The fact that the simplistic goal test used in search does not allow for the identification of milestones.

    (d) The lack of a notion of cost for solutions.

    (e) The difficulty in defining appropriate search heuristics for a given problem domain.

12. Assume actions D and E both make postcondition p true and you are given a plan with existing causal links $A \stackrel{\neg p}{\rightarrow} B$ and $B \stackrel{\neg p}{\rightarrow} C$. Which of the following total orderings would resolve all potential conflicts that might arise from the addition of D and E?

    (a) $A \prec D \prec E \prec B \prec C$

    (b) $A \prec B \prec D \prec C \prec E$

    (c) $A \prec B \prec D \prec E \prec C$

    (d) $A \prec D \prec B \prec C \prec E$

    (e) $A \prec B \prec C \prec D \prec E$

13. You are given four action descriptions with conditional effects:

    $Action(A, \textsc{Precond}{:}\{P, X\}, \textsc{Effect}{:}\{(\textbf{when } Z : \neg X)\})$
    $Action(B, \textsc{Precond}{:}\{\neg Y\}, \textsc{Effect}{:}\{(\textbf{when } P : Z, X)\})$
    $Action(C, \textsc{Precond}{:}\{Z\}, \textsc{Effect}{:}\{(\textbf{when } P : \neg Y), X\})$
    $Action(D, \textsc{Precond}{:}\{\neg X\}, \textsc{Effect}{:}\{(\textbf{when } P : Z, X)\})$

    What state would result from executing the action sequence $[D, C, B, A]$ in the state $\{P, Y\}$?

    (a) The plan isn't executable because the preconditions for action A aren't met.

    (b) $\{P, Z\}$

    (c) $\{P, X, Z\}$

    (d) $\{P, \neg X, \neg Y, Z\}$

    (e) $\{P, X, Y, Z\}$

14. Consider the following PDDL action description in a Blocks World application:

$Action(Move(b, x, y)$,
PRECOND:$On(b, x) \wedge Clear(b) \wedge Clear(y)$
EFFECT:$On(b, y) \wedge Clear(x) \wedge \neg On(b, x) \wedge \neg Clear(y))$

and the state S_1 defined as follows:

S_1 : $\{On(A, B), On(B, C), Clear(A), Clear(D)\}$

Which of the following statements is incorrect?

(a) $Move(A, B, D)$ can be applied in S_1

(b) Applying $Move(A, B, D)$ in S_1 would result in state $\{On(A, D), \neg On(A, B), \neg Clear(D), Clear(B)\}$

(c) $Move(B, C, D)$ cannot be applied in S_1

(d) $[Move(A, B, D), Move(B, C, A)]$ can be applied in S_1

(e) You need a sequence of at least two actions to achieve the goal $On(B, D)$.

15. Assume a planning environment characterised by a manageable list of foreseeable contingencies but also with the possibility of unforeseen failures. Which of the following strategies to cope with this kind of indeterminacy would seem most appropriate?

(a) Sensorless (conformant) planning that will work under any circumstances

(b) Offline conditional planning for all possible contingencies

(c) There is no reasonable way of dealing with this kind of indeterminacy

(d) Offline conditional planning for the foreseeable contingencies combined with online re-planning to handle unexpected failures

(e) Constant replanning so that the agent copes with all circumstances

16. Which of the following statements is correct in the general case, (i.e. if no assumptions regarding conditional independence are made)?

(a) $P(b|a) = \frac{P(a \wedge b)}{P(b)}$

(b) $P(a \wedge b) = P(a|b)P(b|a)$

(c) $P(a \wedge b) = P(a)P(b)$

(d) $P(b|a) = \frac{P(a|b)P(b)}{P(a)}$

(e) $P(a|b, c) = P(a|b)P(c)$

17. Assume the following inhibition probabilities between Boolean cause variables *Rich*, *Smart*, *Beautiful* and Boolean effect variable *Happy*:

$$P(\neg happy|rich, \neg smart, \neg beautiful) = 0.2$$
$$P(\neg happy|\neg rich, smart, \neg beautiful) = 0.5$$
$$P(\neg happy|\neg rich, \neg smart, beautiful) = 0.5$$

What is the probability $P(happy|rich, smart, \neg beautiful)$ assuming that the conditional probabilities of *Happy* are computed using a noisy-OR relation?

(a) 0.05

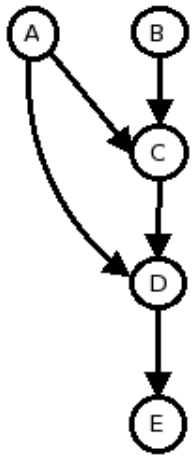
(b) 0.9

(c) 0.25

(d) 0.4

(e) 0.1

18. Consider the following Bayesian network for random variables A, B, C, D and E.



Which of the following is equal to the joint probability distribution $\mathbf{P}(C, D, E)$?

(a) $\mathbf{P}(E|D)\mathbf{P}(D|C, A)\mathbf{P}(C|B, A)\mathbf{P}(B)\mathbf{P}(A)$

(b) $\mathbf{P}(E|A, B, C, D)\mathbf{P}(D|C, B, A)\mathbf{P}(C|B, A)\mathbf{P}(B)\mathbf{P}(A)$

(c) $\mathbf{P}(E|D)\sum_a \mathbf{P}(D|C, a)\sum_b \mathbf{P}(C|b, a)P(b)P(a)$

(d) $\sum_a \sum_b \mathbf{P}(E|D, C, b, a)\mathbf{P}(D|E, C, b, a)\mathbf{P}(C|E, D, b, a)$

(e) None of the above

19. The following formula describes the "forward" part of the algorithm used for prediction in temporal probabilistic models:

$$f_{1:t+1} = \alpha \mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1}) \sum_{\mathbf{x}_t} \mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t})$$

Which of the following statements is incorrect?

(a) α is the normalisation factor

(b) $P(\mathbf{x}_t|\mathbf{e}_{1:t})$ is the current state distribution

(c) $\mathbf{P}(\mathbf{x}_{t+1})$ depends on both $\mathbf{P}(\mathbf{x}_t)$ and $\mathbf{P}(\mathbf{e}_t)$

(d) $\mathbf{P}(\mathbf{e}_{t+1}|\mathbf{X}_{t+1})$ is obtained from the sensor model

(e) $\mathbf{P}(\mathbf{X}_{t+1}|\mathbf{x}_t)$ is obtained from the transition model

20. Let strict preference be denoted by the relation symbol $\succ$, indifference by $\sim$, weak preference by $\succsim$, and lotteries be written as $[p_1, O_1; \ldots; p_n, O_n]$, where each p_i is the probability associated with outcome O_i. Which of the following is not an axiom of utility theory?

(a) $(A \succ B) \vee (B \succ A) \vee (A \sim B)$

(b) $(A \succ B) \wedge (B \succ C) \Rightarrow (A \succ C)$

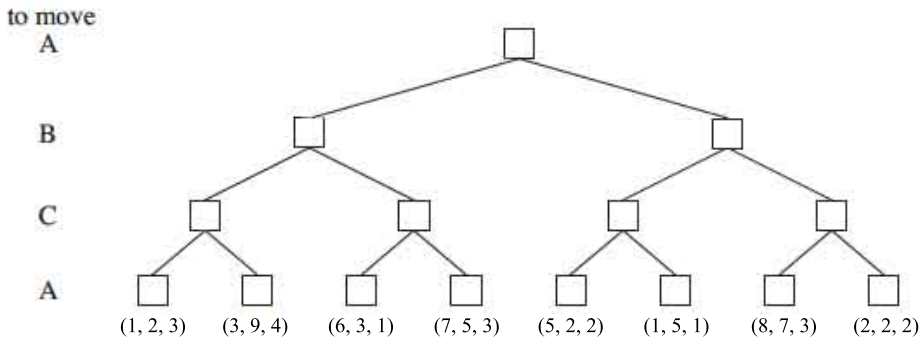
(c) $A \succ B \succ C \Rightarrow \exists p\, [p, A; 1 - p, C] \sim B$

(d) $A \sim B \Rightarrow [p, A; 1 - p, C] \succ [p, B; 1 - p, C]$

(e) $A \succ B \Rightarrow (p \geq q \Leftrightarrow [p, A; 1 - p, B] \succsim [q, A; 1 - q, B])$

Part B
ANSWER ONE QUESTION FROM PART B

1. Consider the following look-ahead tree for a three-player game with players A, B, and C. For each terminal node, a vector (v_A, v_B, v_C) gives the utility of the associated state from each of the player's viewpoint. For example, the leftmost leaf-node of the tree has the following utility vector $(v_A = 1, v_B = 2, v_C = 3)$. Each *level* of the tree is labelled with the corresponding player's turn to move e.g. at the second level, it is player B's turn.



(a) Explain how the standard minimax algorithm can be extended to provide an optimal decision for a three-player game and illustrate your answer using the tree above.

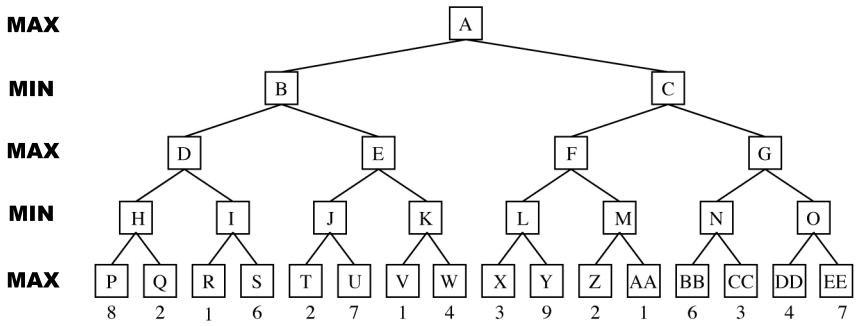
Specify the scores assigned to each of the non-terminal nodes and indicate which move is recommended for A by drawing an arrow to the move's destination node (or by drawing a circle around its destination node).

[8%]

(b) Give the α/β search algorithm.

[7%]

(c) Consider the following 2-player game tree, where each node is named as indicated and MAX and MIN take turn to play (starting with MAX at the root). The number below each leaf node is the result of the evaluation function applied to that leaf.



Which nodes will be pruned from the tree by α/β search when it is searched depth-first, left to right? Assuming a perfect ordering of the leaves is possible, which nodes would be pruned?

In each case, you may give a diagrammatic answer that clearly shows the pruned nodes/branches of the corresponding tree.

[6%]

(d) Iterative deepening is often used in conjunction with α/β search. Briefly explain why this may be advantageous.

[4%]

2. (a) Give an interpretation that *satisfies* the statement

$$\forall x.\forall y.\forall z.\ x + (y + z) = (x + y) + z$$

but falsifies

$$\forall x.\forall y.\ x + y = y + x$$

[5%]

(b) Define the full-resolution rule of inference.

[3%]

(c) Consider the following sentences about a binary relation p:

Irreflexive:	$\forall x.\ \neg p(x, x)$
Transitive:	$\forall x.\forall y.\forall z.\ p(x, y) \wedge p(y, z) \rightarrow p(x, z)$
Asymmetric:	$\forall x.\forall y.\ p(x, y) \rightarrow \neg p(y, x)$

Give a resolution proof that if the binary relation p is irreflexive and *not* asymmetric then it is *not* transitive.

[6%]

(d) Give an algorithm for forward-chaining using Generalised Modus Ponens.

[5%]

(e) Describe two possible ways in which the efficiency of the forward-chaining algorithm from (2d) can be improved. In each case, you need to explain briefly what the issue is and how the improvement addresses it.

[6%]

Part C
ANSWER ONE QUESTION FROM PART C

1. **Planning**

A patient arrives at the doctor's office, with symptoms that could have been caused either by dehydration or by disease D (but not both). The doctor wants to cure the patient of whatever ails him—i.e., the disease, or dehydration. There are two possible actions: *Drink*, which unconditionally cures dehydration, and *Medicate*, which cures disease D but has an undesirable side effect if taken when the patient is dehydrated.

(a) Design a PDDL vocabulary for expressing the above planning problem, and in particular the possible states in this domain.

[5%]

(b) Using your vocabulary, express the doctor's initial knowledge state and his goal, according to the planning problem described above.
Hint: express a knowledge state as a set of states.

[4%]

(c) Using the vocabulary, write the PDDL action schemata for *Drink* and *Medicate*.

[4%]

(d) Write a sensorless plan that solves the planning problem; demonstrate that it works by giving the doctor's knowledge state after each action.
Note: You don't have to identify for each state in the doctor's knowledge state which prior possible state(s) caused it.

[6%]

(e) Suppose that the doctor can sense whether the patient is dehydrated. However, performing *Drink* can fail to hydrate the patient.

i. Re-write the PDDL action schema *Drink*, taking into account its nondeterministic effects.

[2%]

ii. Write a contingent plan that solves the planning problem, given this new action schema *Drink*.

[4%]

2. **Bayesian Network**

Let B_x be a Boolean random variable, with values b (x has blue eyes) and $\neg b$ (x doesn't have blue eyes). Eye colour is inherited via a gene G_x, that also takes values b and $\neg b$. It turns out that for any individual x, G_x and B_x are very likely to have the same value: assume that the probability that $G_x = B_x$ is s for any individual x. Moreover, for any individual x, if $G_x = B_x$, then these variables are equally likely to have the value b as $\neg b$. For any individual x, its gene G_x is very likely to have the same value as one of its parent genes: i.e., where $f(x)$ and $m(x)$ are respectively the father and mother of x, it is very likely that G_x has the same value as $G_{f(x)}$ and/or $G_{m(x)}$. If $G_{f(x)} \neq G_{m(x)}$, then G_x is just as likely to have the value of $G_{f(x)}$ as $G_{m(x)}$. And if $G_{f(x)} = G_{m(x)}$, then there is a small probability that mutation occurs so that G_x takes the value of neither parent—this occurs with probability r.

(a) Draw a Bayesian network containing Boolean variables G_x, $G_{f(x)}$, $G_{m(x)}$, B_x, $B_{f(x)}$ and $B_{m(x)}$ that captures the dependencies in the above description.

[7%]

(b) Assuming that $P(G_{f(x)} = b) = P(G_{m(x)} = b) = p$, derive the conditional probability distribution tables in your Bayesian network for the following variables:

i. $B_{f(x)}$

[10%]

ii. G_x

[8%]

Your answers should be given in terms of p, r and s, and you should explicitly show each step of the derivation with justifications for the step where appropriate.

Specimen Answers

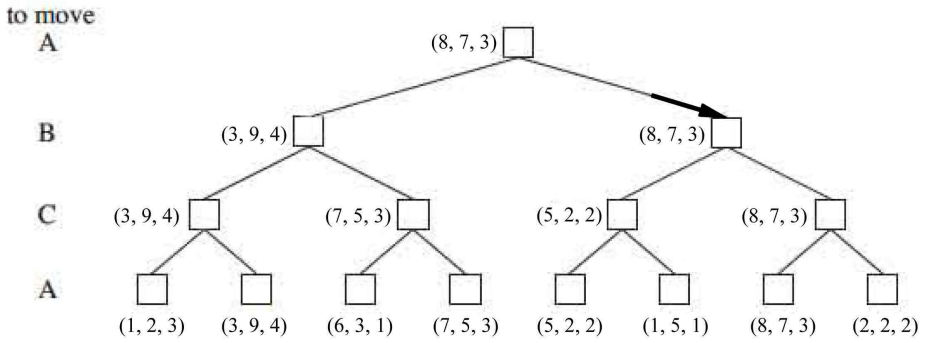
Part A

1. c
2. d
3. e
4. b
5. a
6. c
7. a
8. b
9. c
10. e
11. d
12. e
13. b
14. b
15. d
16. d
17. b
18. c
19. c
20. d

Part B

1. (a) (Partly bookwork) In this example, just as for the terminal nodes, a vector (v_A, v_B, v_C) is attached to each non-terminal ones. In order to assign to the non-terminal nodes, the extended minimax algorithm will progressively propagate vectors up the tree from the leaves to the root (just as the standard minimax propagates single scores up). In this case, though, for any node n, the backed-up value is always the utility vector of the successor state with the highest value for the *player choosing* at n. For instance, at the leftmost state when it is C's turn to move, there are two choices leading to terminal states $(v_A = 1, v_B = 2, v_C = 3)$ and $(v_A = 3, v_B = 9, v_C = 4)$. Since 4 is greater than 3, C will thus choose the second move.

   For the given tree, the scores are inherited as indicated below and the recommended move is as indicated by the arrow.



   Marking guide: 1 mark for explaining that the scores are inherited up the tree. 4 marks for explaining how the backed-up values are computed in general *and* with reference to the given tree. 2 mark for specifying the vectors attached to each non-terminal node and 1 mark for identifying the move that the process recommends.

   (b) (Bookwork) From Russell & Norvig:

```
function ALPHA-BETA-SEARCH(state) returns an action
  v ← MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v

function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v
    α ← MAX(α, v)
  return v

function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← +∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
    if v ≤ α then return v
    β ← MIN(β, v)
  return v
```

Marking guide: 1 marks for first function, with correct initial values for alpha and beta. 3 marks for MAX-VALUE and 3 marks for MIN-VALUE. Note: The algorithm does not have to be given exactly as above. Any reasonable answer describing all the important aspects is acceptable.

(c) The following would be pruned in the first case: node S; the branch consisting of K, V, W; node AA; and and the branch consisting of O, DD, EE i.e the following nodes: S, K, V, W, AA, O, DD, EE.

With perfect ordering, the leaves would be re-arranged in value order, and following nodes would be pruned: Z, N, CC, BB, P, S, E, J, T, U, K, V, W.

Marking guide: 3 marks each.

(d) (Bookwork) One advantage on iterative-deepening is that searches at lower depths can be used to work out move ordering information. The best path at any depth d can be recorded and used to inform move ordering at depth $d + 1$. This acts as a dynamic move-ordering scheme that tries the best moves found in the past and can help prune the search tree, getting it close

to the theoretical limit $O(b^{m/2})$ for perfect move ordering.

Marking guide: Any answer along the above lines will do. It is essential that move-ordering is mentioned.

2. (a) A possible interpretation: Domain is that of lists and + is mapped to the append function.

Marking guide: 1 mark for an interpretation satisfying the first sentence. 1 mark for an interpretation falsifying the second sentence. 3 additional marks if it the *same* interpretation for both cases.

Note: Any other interpretation is (obviously) acceptable.

(b) (Bookwork) The full-resolution rule of inference:

$$\frac{C \vee P_1 \vee \ldots \vee P_m \qquad D \vee \neg P'_1 \vee \ldots P'_n}{(C \vee D)\theta}$$

where θ is the most general unifier (MGU) of the P_is and P'_is.

Marking guide: Deduct 2 marks for any significant errors and 1 mark for minor ones. Only giving binary resolution should be considered a significant error.

(c) The goal (*not* Transitive) is first negated, giving back the original statement i.e. Transitive, and added to Irreflexive and the *negation* of Asymmetric. These sentences are next converted to clausal normal form, and a refutation attempted using resolution.

Negated goal:

$$\neg\neg \text{ Transitive} = \text{Transitive} = \forall x.\forall y.\forall z.\ p(x,y) \wedge p(y,z) \rightarrow p(x,z)$$

Sentences converted to CNF, with variables standardized apart, and Skolem constants a, b:

$$\begin{array}{rl} \text{Irreflexive:} & \neg p(x,x) \\ \text{Transitive:} & \neg p(u,v) \vee \neg p(v,w) \vee p(u,w) \\ \neg\text{Asymmetric:} & p(a,b) \text{ and } p(b,a) \end{array}$$

Proof (using binary resolution):

1. $\neg p(x, x)$
2. $\neg p(u, v) \vee \neg p(v, w) \vee p(u, w)$
3. $p(a, b)$
4. $p(b, a)$
5. $\neg p(x, v) \vee \neg p(v, x)$ Resolve(1,2), $\{u/x, w/x\}$
6. $\neg p(b, a)$ Resolve(3,5), $\{x/a, v/b\}$
7. $\square$ Resolve(4,6), $\{\}$

Marking guide:

1 mark for negating goal. 2 marks for conversion to CNF. 3 marks for proof. Deduct 1 mark if the variables are not standardized apart. Deduct 1 mark if Skolem constants are not mentioned explicitly.

(d) (Bookwork) The following algorithm is an acceptable answer:

function FOL-FC-ASK(KB, α) **returns** a substitution or *false*
  inputs: KB, the knowledge base, a set of first-order definite clauses
    α, the query, an atomic sentence
  local variables: *new*, the new sentences inferred on each iteration

  repeat until *new* is empty
    new $\leftarrow \{\}$
    for each *rule* **in** KB **do**
      $(p_1 \wedge \ldots \wedge p_n \Rightarrow q) \leftarrow$ STANDARDIZE-VARIABLES(*rule*)
      for each θ such that SUBST($\theta, p_1 \wedge \ldots \wedge p_n$) = SUBST($\theta, p'_1 \wedge \ldots \wedge p'_n$)
          for some $p'_1, \ldots, p'_n$ in KB
        $q' \leftarrow$ SUBST(θ, q)
        if q' does not unify with some sentence already in KB or *new* **then**
          add q' to *new*
          $\phi \leftarrow$ UNIFY(q', α)
          if ϕ is not *fail* **then return** ϕ
    add *new* to KB
  return *false*

Marking guide: Note that the algorithm does not need to be given exactly as above. Alternative answers not in pseudo-code but describing the main aspects are acceptable.

(e) (Bookwork) Any two of the following three improvements are acceptable:

- Incremental forward-chaining: In this, at each iteration t, we check a rule only if its premise includes a conjunct p_i that unifies with a fact

p_i newly inferred at iteration $t-1$. The rule-matching step then fixes p_i to match with p'_i from the algorithm in (2d), but allows the other conjuncts of the rule to match with facts from any previous iteration. This algorithm generates exactly the same facts at each iteration as the one in (2d), but is much more efficient.

- Avoid irrelevant facts: Forward chaining makes all allowable inferences based on the known facts, even if they are irrelevant to the goal at hand. One solution is to restrict forward chaining to a selected subset of rules. Another solution, from the field of deductive database, is to rewrite the rule set, using information from the goal, so that only relevant variable bindings–those belonging to a so-called *magic set*-are considered during forward inference.
- Improved conjunct ordering: This involves finding an ordering to solve the conjuncts of the rule premise so that the total cost of matching against facts from the knowledge base is minimized. It turns out that finding the optimal ordering is NP-hard, but good heuristics are available. For example, the minimum-remaining-values (MRV) heuristic used for CSPs can be applied.

Marking guide: 3 marks per improvement. Details must be given i.e. just stating the possible improvements is not enough for full marks.

Part C

1. **Planning**

   (a) There are several alternative vocabularies. Here is one:

   - Terms: p (patient); D (disease); n (nasty side-effects)
   - $has(x, y)$: x has y
   - $dehydrated(x)$: x is dehydrated.

   Marking guideline: Deduct 1 point if they get the type of an expression wrong; deduct 2 points if the vocabulary is too expressive, or not expressive enough. Note, for instance, that you *don't* need to ever refer to the doctor! You also don't have to explicitly refer to the symptoms.

   (b) Initial knowledge state: $\{\{has(p, D)\}, \{dehydrated(p)\}\}$
   Goals state: $\neg has(p, D) \wedge \neg dehydrated(p) \wedge \neg has(p, n)$
   Marking guideline: Don't penalise if they represent the knowledge state as a disjunction of states rather than as a set of states; nor if they represent each state as a conjunction of positive literals rather than as a set of positive literals (though if they do reprsents states this way they may struggle to do part (d) of this question!). But *do* penalise them if they express a state using negation (by 1 point). Penalise them for each inaccurate conjunct in the goal state by 1 point, and each inaccurate state in the initial knowledge state.

   (c)

   $Action(Drink(x),$
   PRECOND:
   EFFECT:$\neg dehydrated(x))$

   $Action(Medicate(x),$
   PRECOND:
   EFFECT:$\neg has(x, D), (\textbf{when}\ dehydrated(x); has(x, n)))$

   (d) $[Drink(p); Medicate(p)]$
   Sequence of knowledge states is:

   i. $\{\{has(p, D)\}, \{dehydrated(p)\}\}$ (initial state)
   ii. $\{\{has(p, D)\}, \{\}\}$ (result of $Drink(p)$)
   iii. $\{\}$ (result of $Medicate(p)$)

Marking Guideline: It is likely that the students will forget that states invoke the CWA, and so they may (wrongly) write negations. Penalise them for this by 1 point. Penalise them further for each incorrect state.

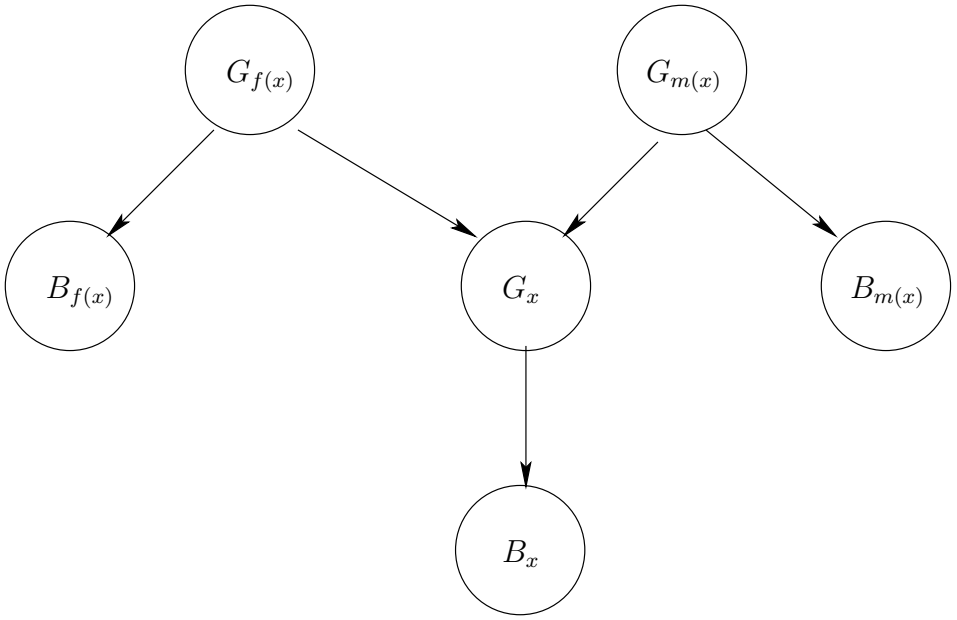
(e) i.

$$\begin{array}{l} \mathit{Action}(\mathit{Drink}(x), \\ \quad \textsc{Precond}: \\ \quad \textsc{Effect}: \mathit{dehydrated}(x) \vee \neg \mathit{dehydrated}(x)) \end{array}$$

ii.

$$\begin{array}{l} [\textbf{if } \mathit{dehydrated}(p) \\ \qquad \textbf{then } (\textbf{while } \mathit{dehydrated}(p); \mathit{Drink}(p)), \\ \qquad \textbf{else } \mathit{Medicate}(p)] \end{array}$$

2. (a) The network should look like this:



Marking Guideline: Deduct one point for each incorrect arrow, and 1 point for each variable that's missing.

(b) i. The conditional probability talbe for $B_{f(x)}$ defines the conditional probability distrubtion $P(B_{f(x)}|G_{f(x)})$.
We'll calculate $P(B_{f(x)} = b|G_{f(x)} = b)$ and $P(B_{f(x)} = b|G_{f(x)} = \neg b)$. Since $B_{f(x)}$ is booelan, this suffices for giving the whole probability distribution.
By Bayes rule, for any individual y:

$$P(B_y|G_y) = \frac{P(B_y, G_y)}{P(G_y)}$$

So we start by calculating the joint probability distribution $P(B_{f(x)} = b, G_{f(x)} = b)$ (in fact, by the description, this joint probability is the same for all individuals—i.e., for x, $f(x)$ and $m(x)$).

$$\begin{aligned} P(G_{f(x)} = B_{f(x)}) = &\ s \\ = &\ P((G_{f(x)} = B_{f(x)} = b) \vee (G_{f(x)} = B_{f(x)} = \neg b)) \\ = &\ P(G_{f(x)} = B_{f(x)} = b) + P(G_{f(x)} = B_{f(x)} = \neg b) - \\ &\quad P(G_{f(x)} = B_{f(x)} = b, G_{f(x)} = B_{f(x)} = \neg b) \\ = &\ P(G_{f(x)} = B_{f(x)} = b) + P(G_{f(x)} = B_{f(x)} = \neg b) \end{aligned}$$

By the problem description, $P(G_{f(x)} = B_{f(x)} = b|G_{f(x)} = B_{f(x)}) = P(G_{f(x)} = B_{f(x)} = \neg b|G_{f(x)} = B_{f(x)})$. And by Bayes' rule:

$$\begin{aligned} P(G_{f(x)} = B_{f(x)} = b|G_{f(x)} = B_{f(x)}) = &\ \frac{P(G_{f(x)}=B_{f(x)}=b, G_{f(x)}=B_{f(x)})}{P(G_{f(x)}=B_{f(x)})} \\ = &\ \frac{P(G_{f(x)}=B_{f(x)}=b)}{P(G_{f(x)}=B_{f(x)})} \end{aligned}$$

By a similar argument:

$$P(G_{f(x)} = B_{f(x)} = \neg b | G_{f(x)} = B_{f(x)}) = \frac{P(G_{f(x)} = B_{f(x)} = \neg b)}{P(G_{f(x)} = B_{f(x)})}$$

So

$$P(G_{f(x)} = B_{f(x)} = b) = P(G_{f(x)} = B_{f(x)} = \neg b)$$

So

$$\begin{aligned} P(G_{f(x)} = B_{f(x)} = b) = & \quad P(G_{f(x)} = B_{f(x)} = \neg b) \\ = & \quad \tfrac{s}{2} \end{aligned}$$

So

$$\begin{aligned} P(B_{f(x)} = b | G_{f(x)} = b) = & \quad \tfrac{P(G_{f(x)} = B_{f(x)} = b)}{P(G_{f(x)} = b)} \\ = & \quad \tfrac{s}{2p} \\ P(B_{f(x)} = \neg b | G_{f(x)} = b) = & \quad 1 - P(B_{f(x)} = b | G_{f(x)} = b) \\ = & \quad 1 - \tfrac{s}{2p} \\ P(B_{f(x)} = b | G_{f(x)} = \neg b) = & \quad 1 - P(B_{f(x)} = \neg b | G_{f(x)} = \neg b) \\ = & \quad 1 - \tfrac{P(B_{f(x)} = \neg b, G_{f(x)} = \neg b)}{P(G_{f(x)} = \neg b)} \\ = & \quad 1 - \tfrac{s}{2(1-p)} \end{aligned}$$

The conditional probability table is therefore:

$G_{f(x)}$	$P(B_{f(x)})$
b	$\frac{s}{2p}$
$\neg b$	$1 - \frac{s}{2(1-p)}$

Marking Guideline: Deduct 1 point for each mistake in the derivation, or if a crucial step in the derivation is missing.

ii. The conditional probability table for G_x must define the conditional probability distribution $P(G_x | G_{f(x)}, G_{m(x)})$. If $G_{f(x)} = G_{m(x)} = b$, then $G_x = b$ will occur so long as mutation doesn't occur. So $P(G_x = b | G_{f(x)} = b, G_{m(x)} = b) = (1 - r)$. By symmetry, $P(G_x = \neg b | G_{f(x)} = \neg b, G_{m(x)} = \neg b) = (1 - r)$, so $P(G_x = b | G_{f(x)} = \neg b, G_{m(x)} = \neg b) = r$. Now consider $G_{f(x)} = b$ and $G_{m(x)} = \neg b$. In this case, G_x is equally likely to take the value of either parent. So $P(G_x = b | G_{f(x)} = b, G_{m(x)} = \neg b) = \frac{1}{2}$, and by symmetry $P(G_x = b | G_{f(x)} = \neg b, G_{m(x)} = b) = \frac{1}{2}$. This is summarised in the following table:

$G_{f(x)}$	$G_{m(x)}$	$P(G_x)$
b	b	$(1 - r)$
$\neg b$	b	$\frac{1}{2}$
b	$\neg b$	$\frac{1}{2}$
$\neg b$	$\neg b$	r

Marking Guideline: Deduct 1 point for each incorrect step in the derivation, or if a crucial step in the derivation is missing.