DESK No. | | | |

FAMILY NAME: ______________________

GIVEN NAMES: ______________________

SIGNATURE: ______________________

STUDENT NUMBER: | | | | | | | | |

THE UNIVERSITY OF WESTERN AUSTRALIA

SEMESTER 2, 2019 EXAMINATIONS

Physics, Mathematics & Computing

Department of Computer Science & Software Engineering

CITS3001

Algorithms Agents and Artificial Intelligence

.

This paper contains: 6 Pages **(including title page)**

Time Allowed: **2:00** hours

INSTRUCTIONS:

Answer all questions in the answer book provided
Each question is worth 10 marks.
The total for the paper is 50.

SUPPLIED STATIONERY	**ALLOWABLE ITEMS**
One 20 page answer booklet	

This page has been left intentionally blank

Q1. Definitions

Define and explain the following terms (a one sentence description is sufficient):

(a) Simulated Annealing

1 mark

(b) NP-Complete

1 mark

(c) Dominance (in the context of A* heuristics)

1 mark

(d) Unsupervised Learning

1 mark

(e) Bayesian Network

1 mark

(f) Q-Learning

1 mark

(g) Adaptive Dynamic Programming

1 mark

(h) Entailment

1 mark

(i) Situation Calculus

1 mark

(j) Clobbering (in the context of a partial order planner)

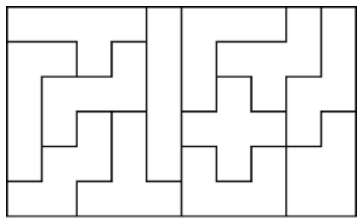
1 mark

Q2. Search

(a) Give the pseudo-code for a *bi-directional breadth first search*, and explain the data structures required for an optimal implementation.

4 marks

(b) Pentominoes are connected shapes that are formed by five equal size squares that have at least one edge in common. There are 12 of these up to reflection and rotation. A well known puzzle, is given some starting arrangement of the twelve pentominoes, add the remaining pentominoes to completely fill a 10 by 6 grid, as in the picture.



Name an *uninformed search algorithm* to solve this problem, and justify your answer if terms of time and space complexity.

3 marks

(c) Suppose we are using the *A* algorithm* to solve the pentominoes puzzle. Propose a heuristic for the algorithm and show that it is admissible and monotonic.

3 marks

Q3. Game-playing

(a) Suppose that you are playing *tic tac toe* (naughts and crosses) with a player (naughts) who you know is playing a random strategy. That is, for every move they will select a random empty cell and place a naught in it.

Describe this problem as a *sequential decision problem*. You do not need to describe every state in the game, but using the above board as an example, describe how states are represented, what actions and transitions are possible, and what costs are assigned to moves and states.

4 marks

(b) Explain the *Bellman equation* in the context of your answer to Part (a) of this question, and describe how it can be used to generate an optimal strategy when versing the random strategy.

3 marks

(c) How will the strategy generated in Part (b) of this question compare with a strategy generated by the *minimax algorithm with α-β pruning*? Which would give the best results when playing the random player?

3 marks

Q4. Learning agents

(a) The CITS3001 project this semester featured the game *LoveLetter,* and it is assumed that you are familiar with the game. Suppose that we have observed someone playing and observed all the times they have had to choose between playing a handmaid and a guard, based on a number of features we consider relevant. From the data collected below, describe how to generate a *decision tree* consistent with the data. Explain the order different features were selected for branching points, and why they were selected.

Cards left	Other players card known	Other player knows you have the handmaid	Players remaining	Action
8	Yes	No	4	Guard
5	No	Yes	2	Handmaid
2	No	Yes	2	Handmaid
3	No	No	3	Handmaid
3	Yes	No	2	Guard
5	No	No	3	Handmaid
1	No	No	4	Guard
4	Yes	Yes	2	Guard
6	No	No	2	Guard

5 marks

(b) *Pig* is a simple dice game. A player rolls a six sided die as many times as they like, and the numbers they roll are added to their score, unless they roll a 1, in which case their score is 0, and their game is over. They can choose to stop rolling at any time, and their aim is to maximize their score. Describe how *temporal difference learning* could be used to learn a strategy for this game and give a simple demonstration, assuming a player rolls: 2,3,2 and then stops for a score of 7.

5 marks

Q5. Reasoning and Planning

(a) Suppose that we have a set of stackable cups, so that one cup can be placed inside the other if and only if it is a smaller cup. We are able to describe the current state of the system using the predicates:

- *inside(X,Y)* to mean cup *Y* is directly inside cup *X*.
- *smaller(X,Y)* to mean cup *Y* is smaller than cup *X*.

Explain the following sentences in plain English:

1. $\forall X\ \neg inside(X,X)$
2. $\forall X\ \forall Y\ \forall Z\ (smaller(X,Y) \land smaller(Y,Z)) \rightarrow smaller(X,Z)$
3. $\forall X\ (\exists Y\ smaller(Y,X) \rightarrow \exists Y\ inside(Y,X))$

3 marks

(b) Given the predicates in Part (a), and some constant *A* (a cup) assume we have a knowledge base consisting of the formulas:

- $\exists X\ inside(X,A)$
- $\forall X\ \forall Y\ inside(X,Y) \rightarrow smaller(X,Y)$

Use resolution to show that this knowledge base entails $\exists X\ smaller(X, A)$

3 marks

(c) Given the predicates in Part (a), suppose we have a planning system with actions:

Action	Precondition	Effect
Put_In(X,Y)	$\forall Z \neg inside(X, Z) \land smaller(X,Y) \land \forall Z \neg inside(Z,Y)$	*inside(X, Y)*
Take_Out(X,Y)	*inside(X,Y)*	$\neg inside(X, Y)$

Describe the main steps for a partial order planner to construct a plan to go from an initial state where cups *B* and *C* are empty, and cup *C* is in cup *A*, to a goal state where cup *B* is in cup *A*, and cup *C* is in cup *B*. You may assume the predicates *smaller(A,B), smaller(B,C)* and *smaller(A,C)* are all true.

4 marks

END OF PAPER