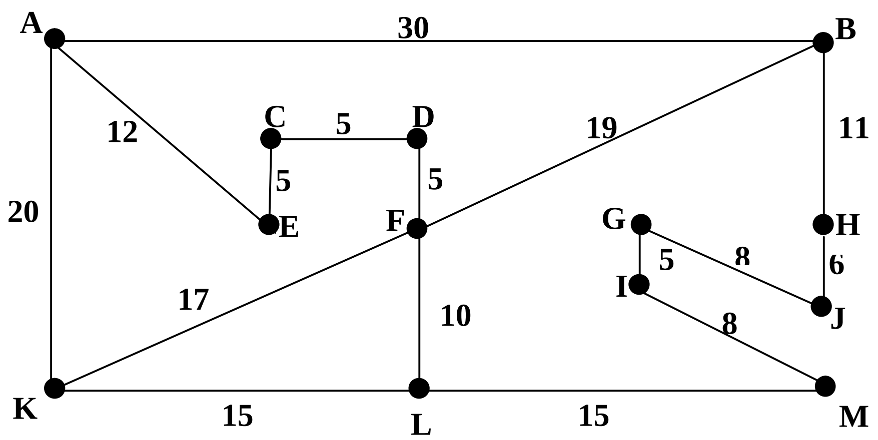
# G5AIAI – Introduction to AI Exam
# 2004/2005

The rubric for this examination should read as follows

*Marks will be awarded for the first FOUR questions in the answer book. Clearly cross through any questions that you do NOT wish to be considered and ensure you state on the front of the answer book the FOUR questions you have attempted.*

## Question 1

Consider the following map (not drawn to scale).




Use the A* algorithm to work out a route from town A to town M. Use the following cost functions.

- G(n) = The cost of each move as the distance between each town (shown on map).
- H(n) = The Straight Line Distance between any town and town M. These distances are given in the table below.

**Straight Line Distance to M**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A | 56 | E | 29 | I | 8 | M | 0.00 |
| B | 22 | F | 30 | J | 5 | | |
| C | 30 | G | 14 | K | 30 | | |
| D | 29 | H | 10 | L | 15 | | |

a) Provide the search tree for your solution, showing the order in which the nodes were expanded and the cost at each node. You should not re-visit a town that you have just come from.

(14 marks)

State the route you would take and the cost of that route.

(2 marks)

b) Assume the estimated costs by the heuristic were replaced and shown in the following table

**Straight Line Distance to M**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| A | 80 | E | 10 | I | 20 | M | 0.00 |
| B | 10 | F | 30 | J | 50 | | |
| C | 50 | G | 60 | K | 60 | | |
| D | 20 | H | 30 | L | 20 | | |

What route would now be returned by the A* algorithm and what would the cost of that route be?

(6 marks)

c) Comment on the optimality of the two A* algorithms. How do you account for the different routes returned?

(3 marks)

## Question 2

(a) Nim is a two-player game. The game starts with a single stack of 7 tokens. At each move a player selects one stack and divides it into two non-empty, non-equal stacks. A player who is unable to move loses the game.

Draw the complete search tree for nim.

(10 marks)

(b) Assume two players, min and max, play nim (as described above). **Min plays first.**

If a terminal state in the search tree developed above is a win for min, a utility function of zero is assigned to that state. A utility function of 1 is assigned to a state if max wins the game.

Apply the minimax algorithm to the search tree to assign utility functions to all states in the search tree.

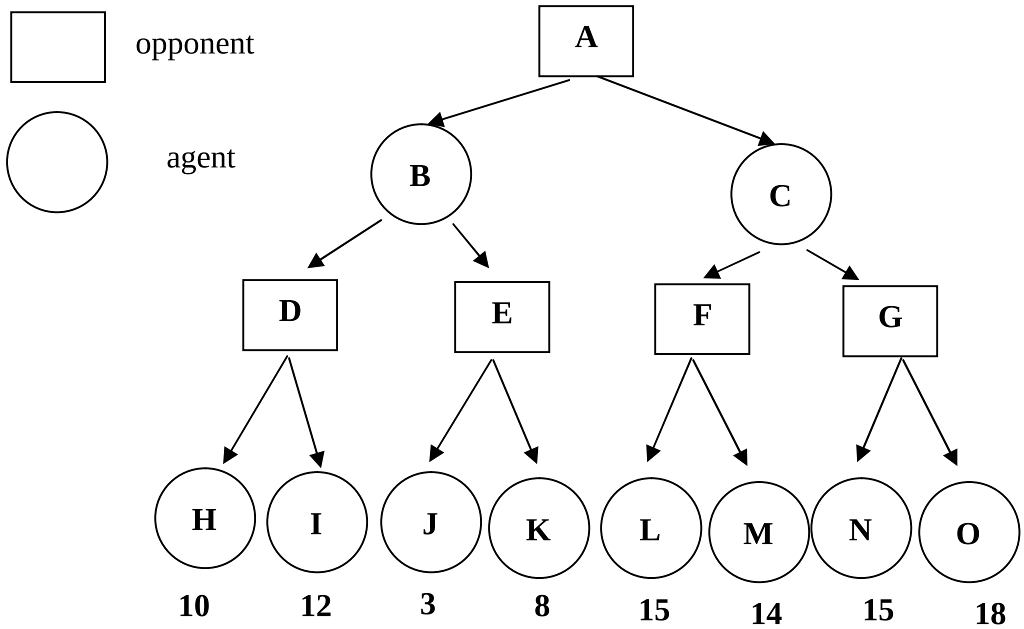
(3 marks)

(c) If both min and max play a perfect game, who will win? Explain your answer.

(3 marks)

(d) Given the following search tree, apply the alpha-beta pruning algorithm to it and show the search tree that would be built by this algorithm. Make sure that you show where the alpha and beta cuts are applied and which parts of the search tree are pruned as a result.

(9 marks)



# Question 3

a) For each of the truth tables below say whether it is possible for a perceptron to learn the required output.

In each case, explain the reason behind your decision.

i)

| | | | | |
|---|---|---|---|---|
| Input | 0 | 0 | 1 | 1 |
| Input | 0 | 1 | 0 | 1 |
| Required Output | 0 | 1 | 1 | 0 |

ii)

| | | | | |
|---|---|---|---|---|
| Input | 0 | 0 | 1 | 1 |
| Input | 0 | 1 | 0 | 1 |
| Required Output | 0 | 0 | 1 | 1 |

iii)

| | | | | |
|---|---|---|---|---|
| Input | 0 | 0 | 1 | 1 |
| Input | 0 | 1 | 0 | 1 |
| Required Output | 1 | 1 | 1 | 1 |

(9 marks)

c) Describe the features of a McCulloch-Pitts neural network.

(5 marks)

b) A perceptron with two inputs has a threshold level set at the point at which it will fire (i.e. output a one). It is sometimes convenient to always set the threshold level to zero.
Show how this can be achieved by describing two perceptrons which act in the same way but one has its threshold set to a non-zero figure and the other perceptron has a zero threshold.

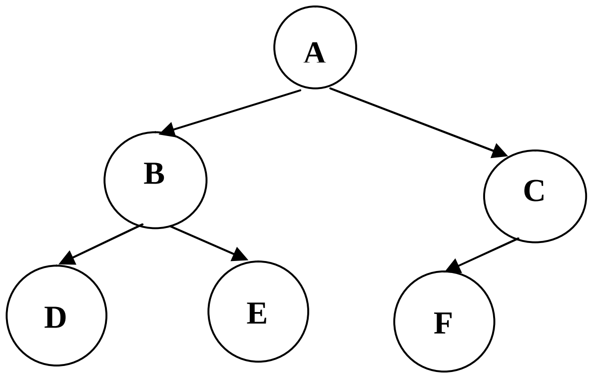
(11 marks)

## Question 4

a) Show how Breath-First-Search and Depth-First-Search can be implemented using some appropriate pseudo-code. If you use another search algorithm as a sub-routine then show this algorithm in detail as well.

(10 marks)

b) For the search tree below, show at each step what nodes are in the queue for both the Breath-First-Search and Depth-First-Search. Show the list of nodes that are expanded.




(8 marks)

c) What is the worst-case time and space complexity of the above two algorithms.

(5 marks)

d) Describe the terms complete and optimal with regards to evaluating search strategies.
Are either Breath-First-Search or Depth-First-Search complete? Is either of them optimal?

(2 marks)

# Question 5

a) Briefly describe the Turing Test

(6 marks)

b) Briefly describe the Chinese Room Experiment.

(6 marks)

c) If the Turing Test is passed does this show that computers exhibit intelligence? State your reasons.

(13 marks)

## Question 6

a) Given the following 8-puzzle, define the problem as a search problem in terms of states, operators, a goal test and a path cost.

| | | |
|---|---|---|
| | 4 | 5 |
| 6 | 1 | 8 |
| 7 | 3 | 2 |

Initial State

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 8 | | 4 |
| 7 | 6 | 5 |

Goal State

(8 marks)

b) Describe the history of checkers (draughts) in the context of computers playing the game.

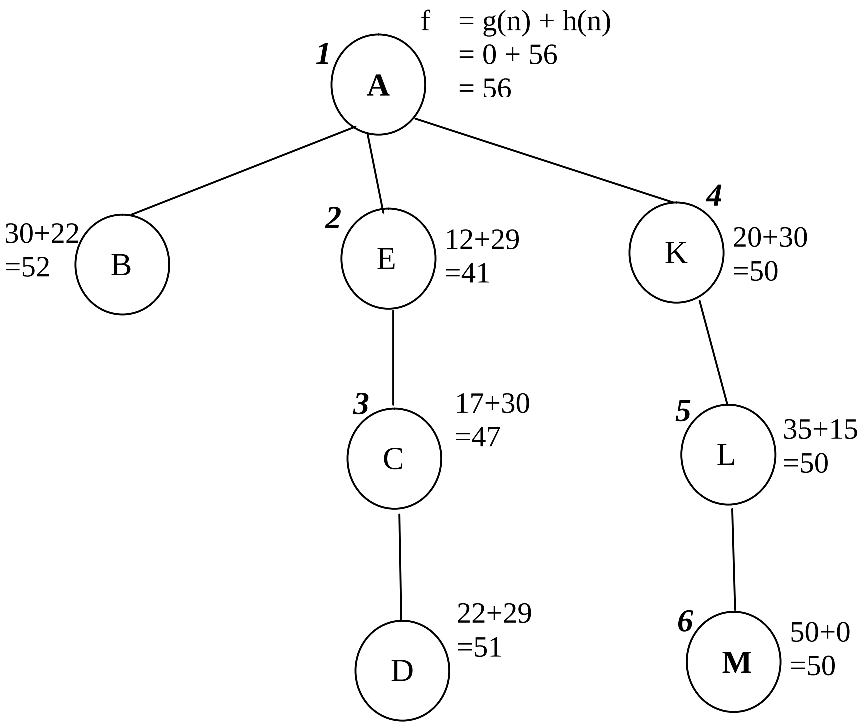
(12 marks)

c) With reference to the Travelling Salesman Problem explain what is meant by combinatorial explosion and what effect this has in finding an optimal solution?

(5 marks)

# Question 1 – Model Answer

**a) The search tree is produced, showing the cost function at each node**
(The numbers in italic show the order in which the nodes were expanded)




I will award marks on the following basis

6 marks : for getting the search tree correct
3 marks : for showing separate (correct) $f(n)$ and $g(n)$ values (i.e. separating the two and showing them correctly)
3 marks : for showing the correct total values for each node
2 marks : for showing the order in which the nodes were expanded (these are shown as the numbers in italic)

The marks will be awarded pro-rata, if applicable.

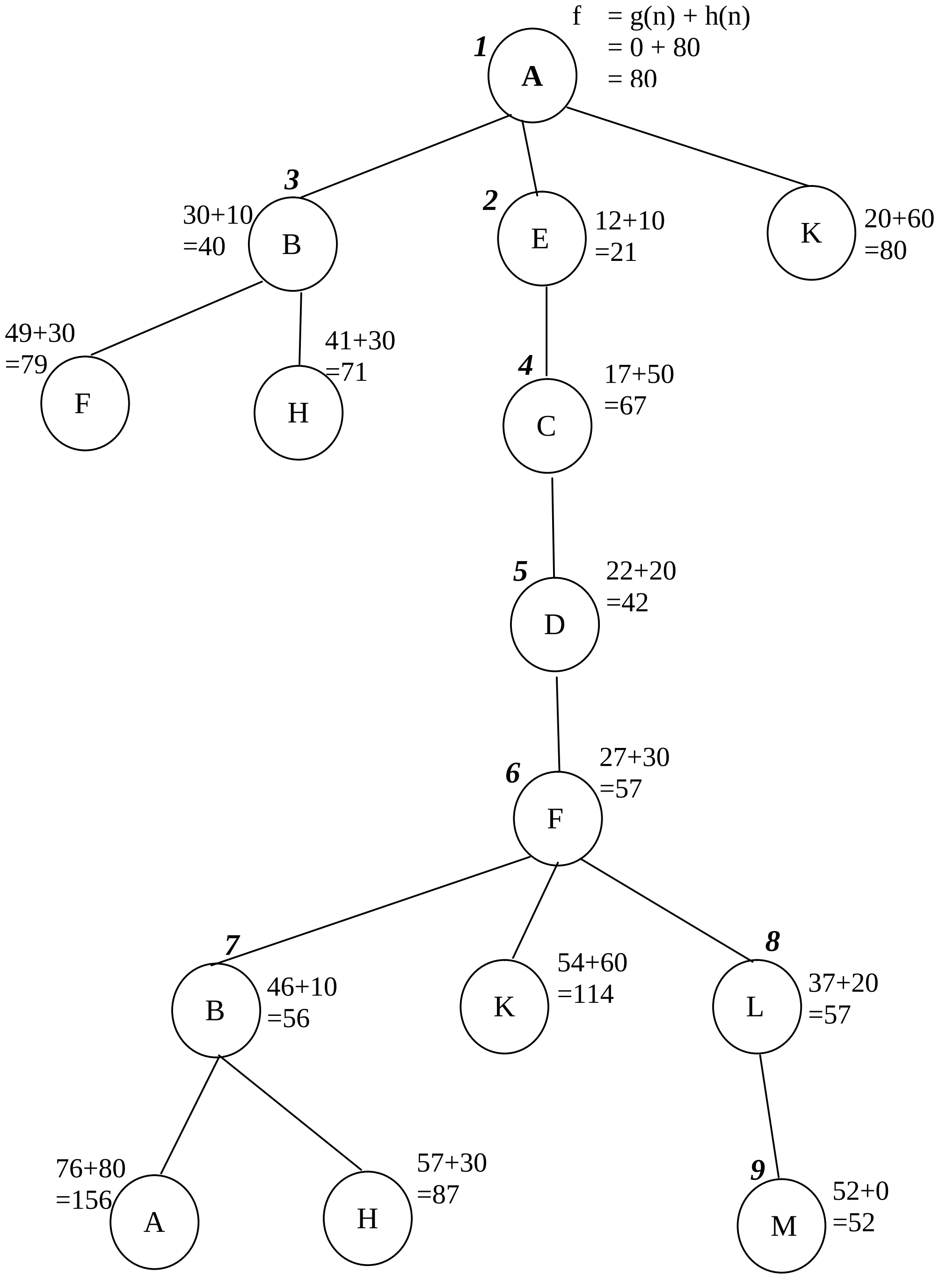
**State the order in which the nodes were expanded**

The route is A, E, C, K, L, M (1 mark)
The cost is 50 (1 mark)

Note that E and C are expanded before K. If this is not done (by providing the answer of A, K, L, M) then the student has not applied the algorithm correctly.

**b) What route would now be returned by the A* algorithm and what would the cost of that route be?**
The answer I am looking for is
The route is A, E, B, C, D, F, B, L, M (1 mark)
The cost is 52 (1 mark)

This part of the question does not ask for the search tree but the students will have to draw it in order to answer the question (the search tree is shown below). Therefore, 4 marks will be awarded for drawing the search tree (the marks will be awarded pro-rata, depending on how much of the tree they get right).




**c) How do you account for the different routes returned by the two A* algorithms?**

I am looking for the following statements from the student.

- The first table was an ***admissible heuristic***, that is one that ***never over-estimates*** the cost to the goal (1 mark)
- The second table was not an admissible heuristic in that some of the values did over-estimate the cost to the goal (1 mark).
- The A* algorithm is only guaranteed to find the optimal solution ***if the heuristic used is admissible*** (1 mark).

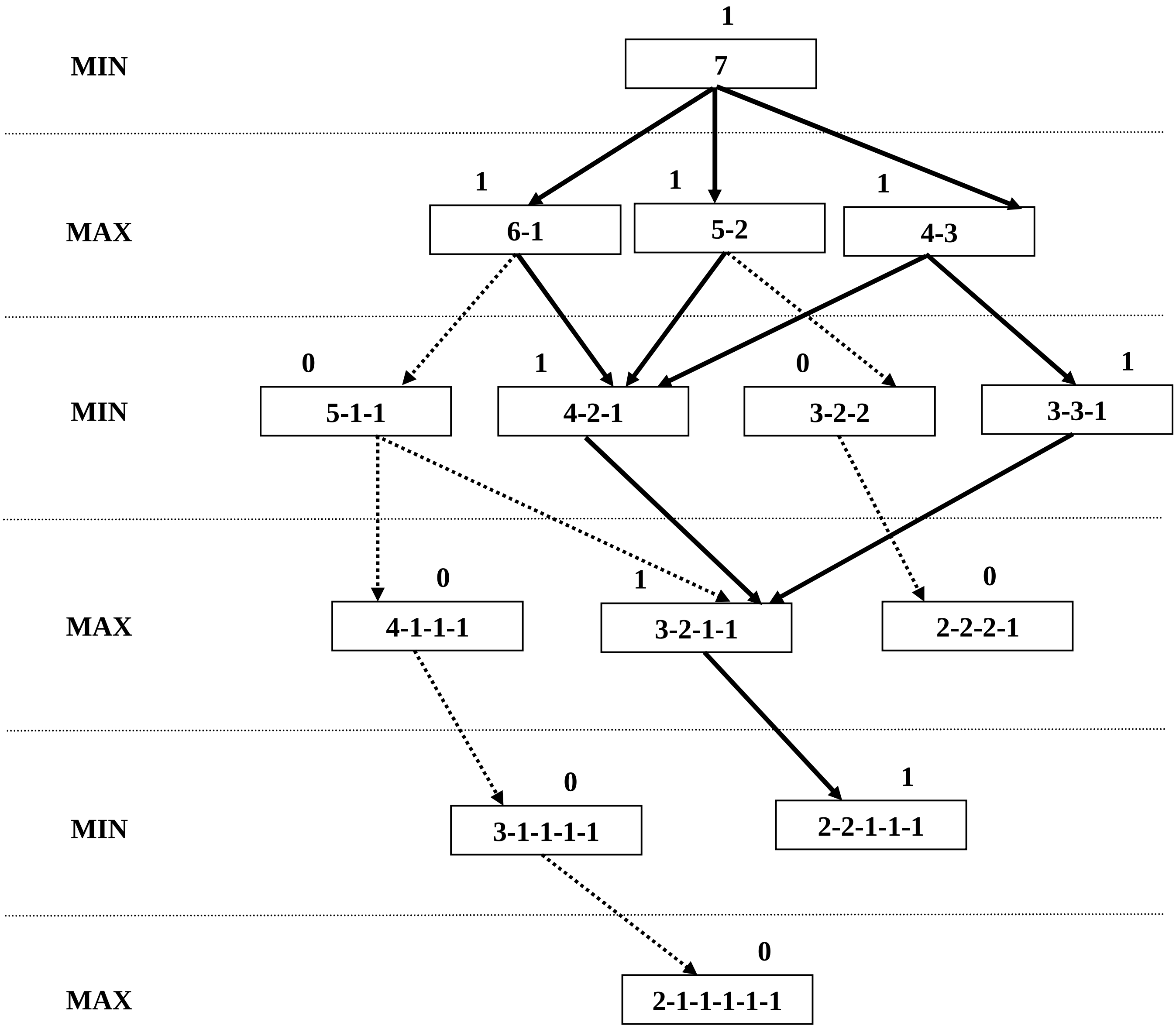
It is ***not*** permissible for the student to simply state "*because the straight line heuristic cost table has changed, the A* algorithm will obviously return a different solution*." I am looking for an understanding of the concept of admissibility.

Note : Although the students have no proof that one heuristic is admissible and the other is not, this is obvious as one heuristic returned the optimal solution and the other did not.

# Question 2 – Model Answer

**(a) Draw the complete search tree for nim.**

The search tree is shown below. The student might draw a search tree which duplicates some of the nodes. This is acceptable.
Note : At this stage, only the search tree is required. The "Min", "Max" and the Utility Functions are not required.




**(b) Apply the minimax algorithm to the search tree to assign utility functions to all states in the search tree.**

The utility functions are shown in the search tree above. The student should initially assign values to the terminal states and then propagate these up the tree, depending on whether it is min or max's turn to move (min minimises the utility function of its child nodes, max maximises).

The student should mark the tree with the players turn – deduct one mark if this is not done. The other two marks are for correctly assigning the correct utility vales to each node. Give one mark if the student appears to be doing it correctly but has made a mistake.

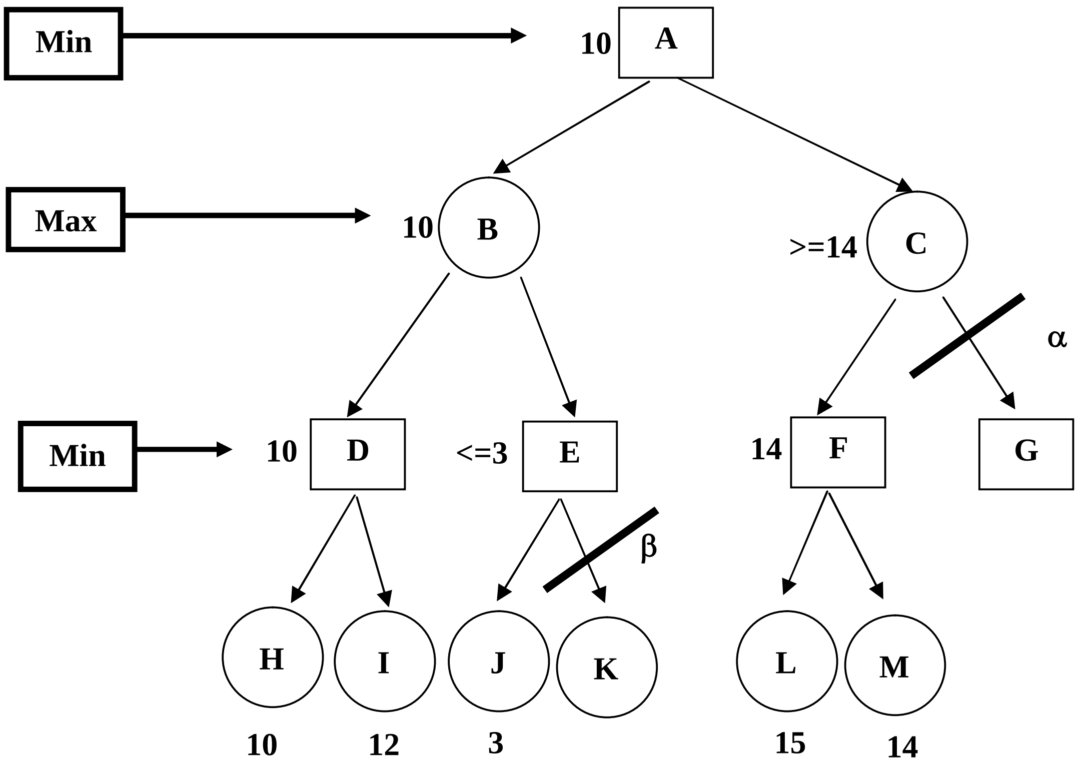
**(c) If both min and max play a perfect game, who will win? Explain your answer and show the path taken by the player that wins.**

As the value at the top of the search tree is 1, it shows that maximise is guaranteed to win (if it plays a perfect game). This is because minimise will never be able to follow a path that leads to a utility function of 0. One mark for pointing this out.

Two marks for showing the path that would be taken, if both players played a perfect game. These are shown in bold in the search tree above.

**(d) Apply the alpha-beta pruning algorithm to it and show the search tree that would be built by this algorithm. Make sure that you show where the alpha and beta cuts are applied and which parts of the search tree are pruned as a result.**

The search tree that the student should re-produce is shown below. If the student produces this search tree they should receive 6 marks (note any extra nodes they produce should be penalised as the idea behind the alpha-beta algorithm is that it restricts the number of nodes that are expanded).
The other three marks are to be allocated based on the way the student explains how the search tree was built.
One mark for stating that alpha-beta pruning ***must*** use Depth First Search.
One mark for explaining why the alpha cut off can be made. One mark why the beta cut off can be made.



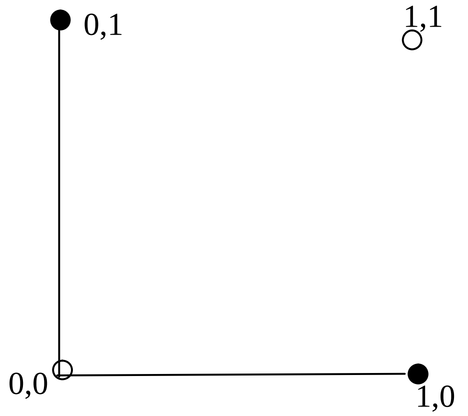
# Question 3 – Model Answer

**a) For each of the truth tables [*not shown in the model answer*] below say whether it is possible for a perceptron to learn the required output.**

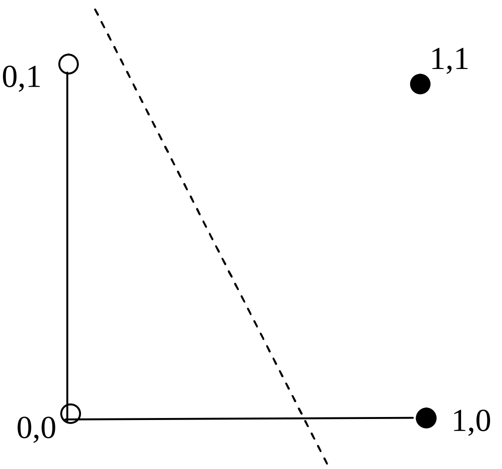
Three marks for each

Only i) cannot be learnt. This is because it is not ***linearly separable***. This can be shown on the diagrams below (where the outputs have been plotted and the filled circles represent a 1 and the hollow circles represent a zero.).
Those problems which are linearly separable can have a line dividing the "1" outputs from the "0" outputs. In the case of i) this is not possible.

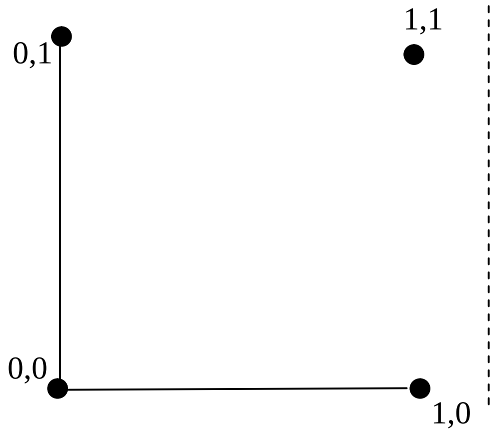
**i)**




**ii)**




**iii)**




**b) Describe the features of a McCulloch-Pitts neural network.**

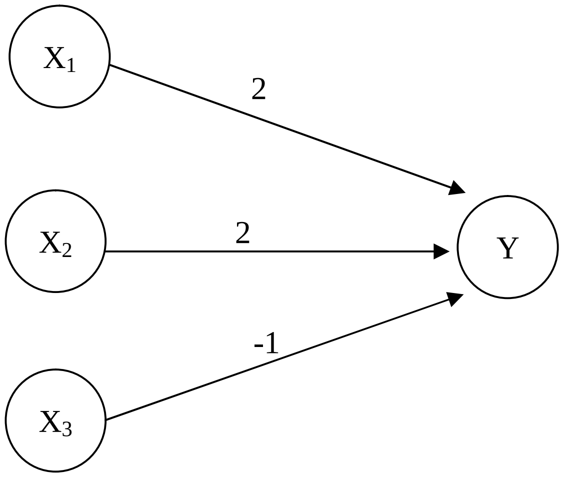
We can make the following statements about a McCulloch-Pitts network

- The activation of a neuron is binary. That is, the neuron either fires (activation of one) or does not fire (activation of zero).
  For the network shown below the activation function for unit *Y* is

  $$f(y\_in) = 1, \text{ if } y\_in >= \theta \text{ else } 0$$

  where *y_in* is the total input signal received
  θ is the threshold for *Y*.
- Neurons is a McCulloch-Pitts network are connected by directed, weighted paths.

- If the weight on a path is positive the path is excitatory, otherwise it is inhibitory.
- All excitatory connections into a particular neuron have the same weight, although different weighted connections can be input to different neurons.
- Each neuron has a fixed threshold. If the net input into the neuron is greater than the threshold, the neuron fires.
- The threshold is set such that any non-zero inhibitory input will prevent the neuron from firing.
- It takes one time step for a signal to pass over one connection.

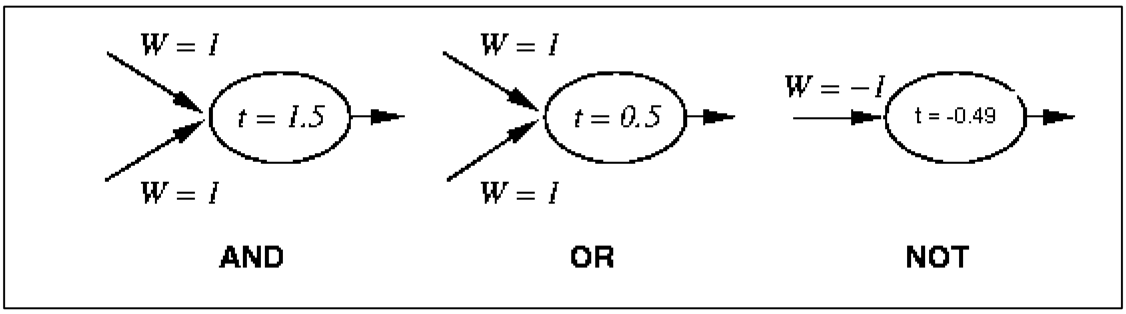



A sample McCulloch-Pitts network is shown above and some of the statements can be observed.
In particular, note that the threshold for Y has equal 4 as this is the only value that allows it to fire, taking into account that a neuron cannot fire if it receives a nonzero inhibitory input.

**c) A perceptron with two inputs has a threshold level set at the point at which it will fire (i.e. output a one).**
**It is sometimes convenient to always set the threshold level to zero.**
**Show how this can be achieved by describing two perceptrons which act in the same way but one has its threshold set to a non-zero figure and the other perceptron has a zero threshold.**

***<u>Perceptrons with two inputs and the threshold non-zero</u>***

A perceptron (with two inputs) to act as a logic gate could be modelled as follows (three examples are shown – students would only need to show one example).




If we consider the AND function we can see that it acts correctly for the four possible inputs (see table below).

Note in the table Sum is defined as (Input 1 * Weight 1) + (Input 2 * Weight 2)

**Step(t) is defined as returning 1 if Sum => t else 0 and the output of Step(t) is also the output (or activation level) of the perceptron.**

## AND

| Input 1 | Input 2 | Weight 1 | Weight2 | Sum | Step(t) |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 |
| 1 | 0 | 1 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 2 | 1 |

Similarly, OR and NOT can be shown as follows

## OR

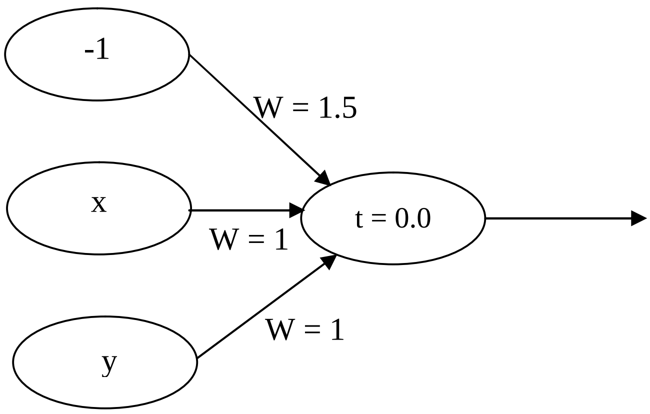
| Input 1 | Input 2 | Weight 1 | Weight 2 | Sum | Step(t) |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 1 | 1 | 0.5 | 1 |
| 1 | 0 | 1 | 1 | 0.5 | 1 |
| 1 | 1 | 1 | 1 | 1 | 1 |

## NOT

| Input 1 | Weight 1 | Sum | Step(t) |
|---|---|---|---|
| 0 | -1 | 0 | 1 |
| 1 | -1 | -0.49 | 0 |

## *Perceptron with three inputs and the threshold set to zero*

It is possible to have an extra input whose activation is set to –1 and the weight from that input unit to the output neuron is set to the required threshold level. Diagramatically this can be shown as follows




It can be shown that this acts in the same way as the previous perceptrons.

To demonstrate the perceptrons act in the same way the following tables are given

**AND (weight on extra neuron = 1.5)**

| Input 1 | Input 2 | Input 3 | Sum | Step(0) |
|---|---|---|---|---|
| -1 | 0 | 0 | -1.5 | 0 |
| -1 | 0 | 1 | -0.5 | 0 |
| -1 | 1 | 0 | -0.5 | 0 |
| -1 | 1 | 1 | 0.5 | 1 |

**OR (weight on extra neuron = 0.5)**

| Input 1 | Input 2 | Input 3 | Sum | Step(0) |
|---|---|---|---|---|
| -1 | 0 | 0 | -0.5 | 0 |
| -1 | 0 | 1 | 0 | 1 |
| -1 | 1 | 0 | 0 | 1 |
| -1 | 1 | 1 | 0.5 | 1 |

**NOT (weight on extra neuron = -0.49)**

| Input 1 | Input 2 | Sum | Step(0) |
|---|---|---|---|
| -1 | 0 | 0.49 | 1 |
| -1 | 1 | -0.51 | 0 |

Again, I would not expect the students to show three examples. They are just shown for completeness.

# Question 4 – Model Answer

**a) Show how BREADTH-FIRST-SEARCH and DEPTH-FIRST-SEARCH can be implemented using some appropriate pseudo-code. If you use another search algorithm as a sub-routine then show this algorithm in detail as well.**

## Breadth-First Search Implementation

**Function** BREADTH-FIRST-SEARCH(*problem*) **returns** a solution or failure
    **Return** GENERAL-SEARCH(*problem*,ENQUEUE-AT-END)

## Depth-First Search Implementation

**Function** DEPTH-FIRST-SEARCH(*problem*) **returns** a solution or failure
    **Return** GENERAL-SEARCH(*problem*,ENQUEUE-AT-FRONT)

## Both searches call the GENERAL-SEARCH algorithm

```
Function GENERAL-SEARCH(problem, QUEUING-FN) returns a solution or failure
   nodes = MAKE-QUEUE(MAKE-NODE(INITIAL-STATE[problem]))
   Loop do
      If nodes is empty then return failure
      node = REMOVE-FRONT(nodes)
      If GOAL-TEST[problem] applied to STATE(node) succeeds then return node
      nodes = QUEUING-FN(nodes,EXPAND(node,OPERATORS[problem]))
   End
End Function
```

**b) For the search tree shown, show at each step what nodes are in the queue for both the Breath-First-Search and Depth-First-Search. Show the list of nodes that are expanded.**
I'm assuming that the nodes at each step are expanded in alphabetic order (i.e. D is expanded before E). The students might do different thing, which is fine here.

Depth First Search

| | | |
|---|---|---|
| A | | |
| B | C | |
| D | E | C |
| E | C | |
| C | | |
| F | | |

Nodes expanded are A, B, D, E, C and F, as shown at the 1st positions in the queues.

Breath First Search

| | | |
|---|---|---|
| A | | |
| B | C | |
| C | D | E |
| D | E | F |
| E | F | |
| F | | |

Nodes expanded are A, B, C, D, E and F, as shown at the 1st positions in the queues.

**c) What is the worst-case time and space complexity of the above two algorithms.**

| Evaluation | Breadth First | Depth First |
|---|---|---|
| Time | $B^D$ | $B^M$ |
| Space | $B^D$ | BM |

| | | | |
|---|---|---|---|
| Where | B | = | Branching Factor |
| | D | = | Depth of Solution |
| | M | = | Maximum Depth of the Search Tree |

2 marks are given to the correct answer for time and space complexity, respectively. 1 mark is given to the note (B, D and M).

**d) Describe the terms complete and optimal with regards to evaluating search strategies? Are either DEPTH-FIRST-SEARCH or BREADTH-FIRST-SEARCH complete? Are either of them optimal?**

Complete : Is the search guaranteed to find a solution if there is one?
Optimal : Is the search guaranteed to find the optimal (cheapest) solution (however cheapest has been defined for that search problem)?

Breadth-First Search is both optimal and complete
Depth-First Search is neither optimal nor complete

# Question 5 – Model Answer

### a) Describe The Turing Test

In this part of the answer I am looking for the "classic" description of The Turing Test (i.e an interrogator asks a computer and a human questions without knowing which is which and than has to decide which is the human and which is the computer).
If the computer can fool the interrogator then it has passed The Turing Test and, according to Turing, exhibits intelligence.

I will give marks for clarity of description (which may include a diagram) and also for some examples from Turing's paper which describes some of the things the computer should be able to do. For example, answer a question on maths but take its time and occasionally get it wrong. Or discuss a sonnet of Shakespeare.

### b) Briefly describe the Chinese Room Experiment

The system comprises a human, who only understands English, a rule book, written in English, and two stacks of paper. One stack of paper is blank. The other has indecipherable symbols on them.
In computing terms the human is the CPU, the rule book is the program and the two stacks of paper are storage devices.
The system is housed in a room that is totally sealed with the exception of a small opening.
The human sits inside the room waiting for pieces of paper to be pushed through the opening. The pieces of paper have indecipherable symbols written upon them.
The human has the task of matching the symbols from the "outside" with the rule book. Once the symbol has been found the instructions in the rule book are followed. This may involve writing new symbols on blank pieces of paper or looking up symbols in the stack of supplied symbols.
Eventually, the human will write some symbols onto one of the blank pieces of paper and pass these out through the opening.

Assume that the symbols passed into the room were valid Chinese sentences, which posed questions. Also assume that pieces of paper passed out of the room were also valid Chinese sentences, which answered those questions. Searle argues that we have a system that is capable of passing the Turing Test and is therefore intelligent according to Turing. But Searle also argues that the system does not understand Chinese as it just comprises a rule book and stacks of paper which do no understand Chinese.
Therefore, according to Searle, running the right program does not necessarily generate understanding.

### c) If The Turing Test is passed does this show that computers exhibit intelligence? State your reasons.

I am looking for the student to come clearly down on one side of the fence or the other. I don't really want an argument put forward from both sides.
I will award marks for how convincing their argument is.
They will get more marks for using some other published work as a backup for their case.
For example, they could quote Searle's Chinese Room and argue that this shows that a machine which exhibits intelligence can still ***not*** be considered intelligent.
They might quote Copeland (in his response to Searle) and say that the system, as a whole understands, and can therefore be considered intelligent.

# Question 6 – Model Answer

**a) Given the following 8-puzzle, define the problem as a search problem in terms of states, operators, a goal test and a path cost.**

**States :** A state description specifies the location of each tile (and the blank) at each location in the state space. The student might define (say) a 3x3 array is used to hold the state that represents the current board position.

**Operators :** For this problem it is worth just having one operator that says that the blank is allowed to move left, right, up or down; with relevant restrictions to stop the blank moving "off the board."

**Goal Test :** Simply check the current state against the goal state that is given

**Path Cost :** Each movement of the blank costs 1 so the total path cost relates to the number of tiles moved.

Marks will be awarded based on how close they have matched the answer above.

**c) Describe the history of checkers (draughts) in the context of computers playing the game.**

This question is really just to see if the student has read and digested the course notes, as a history of the game was given (and presented in the lectures).

Marks will be awarded using the following scheme.

Arthur Samuel, in 1952 (see Samuel, 1963), wrote the first checkers program. The original program was written for an IBM 701 computer. In 1954 he re-wrote the program for an IBM 704 and added a learning mechanism. What makes this program stand out in AI history is that the program was able to learn its own evaluation function. Taking into account the IBM 704 had only 100,00 words of main memory, magnetic tape for long-term storage and a cycle time of almost one-millisecond, this can be seen as a major achievement in the development of AI.
Samuel made the program play against itself and after only a few days play, the program was able to beat its creator and compete on equal terms with strong human opponents.

**4 marks for mentioning Samuel (marks awarded for completeness of above)**

It remains as a testament to Samuel that there was little more work on checkers until Jonathon Schaeffer et. al. developed Chinook (see Schaeffer, 1996). This program uses alpha-beta search (which we consider below) and also has a database to allow it to play a perfect end game. In 1992 Chinook won the US Open and subsequently challenged for the world championship. Dr. Marion Tinsley had been the world champion for over 40 years. In that time she only lost three games. Playing Chinook she lost her fourth and fifth game but ultimately won the match by 21.5 points to Chinook's 18.5 points. In August 1994 there was a re-match but the match ended prematurely when Dr. Tinsley had to withdraw for health reasons. As a result of this Chinook become the official world champion. Scheaffer (1996, p.447) claimed that Chinook was rated at 2814. The best human players are rated at 2632 and 2625. Chinook did not include any learning mechanisms.

**4 marks for mentioning Chinook (marks awarded for completeness of above)**

More recently (Kumar, 2000) developed a checkers program that "learnt" how to play a good game of checkers. The program started knowing just the rules of the game so that it could make legal moves. The program was allowed to evolve by creating a *population* of games that competed against one another, with the best games *surviving* and being adapted in some way before competing again. The adaptation was done using a neural

network with the weights on the synapses being changed by an *evolutionary strategy*. The best program was allowed to compete against a commercial version of checkers and it beat it 6-0. The program got called Anaconda due to the way it put a strangle hold on its opponents.
I actually saw this program as the CEC conference at which the paper was presented. It challenged anybody who cared to play it – and remain undefeated throughout the duration of the conference.

**4 marks for mentioning Kumar (marks awarded for completeness of above)**

**c) With reference to the Travelling Salesman Problem explain what is meant by combinatorial explosion and what effect this has in finding an optimal solution?**

The number of solutions is n! (n factorial), where n is the number of cities. This results in an exponential rise in the number of solutions. For example, for 10 cities the number of possible routes is 3,628,800.
This is known as combinatorial explosion where the number of solutions rises exponentially.
The effect this has with regards to TSP is that is quickly becomes impossible to search the entire search space (i.e. enumerate all possible solutions and choose the best route).
Therefore, heuristic methods are often used to find solutions to these problems.