IMPERIAL COLLEGE LONDON

DEPARTMENT OF ELECTRICAL AND ELECTRONIC ENGINEERING
EXAMINATIONS 2012

EEE/ISE PART III/IV: MEng, BEng and ACGI

ARTIFICIAL INTELLIGENCE

Thursday, 10 May 2:30 pm

Time allowed: 3:00 hours

There are SIX questions on this paper.

Answer FOUR questions.

All questions carry equal marks

Any special instructions for invigilators and information for candidates are on page 1.

Examiners responsible	First Marker(s) :	J.V. Pitt
	Second Marker(s) :	T-K. Kim

The Questions

1 a) Explain the operation of Depth First, Breadth First, and Iterative Deepening Depth First graph search algorithms.

[3]

b) Compare and contrast the performance of the three algorithms with respect to appropriate criteria.

[3]

c) There is a robot standing in front of an infinite 1-dimensional wall, at a position $x = 0$. It is told there exists a single door at some unknown position y along the wall. But it does not know in which direction the door is to be found (i.e., whether $y > 0$ or $y < 0$). The goal of the robot is to find the door.

Formulate, in Prolog or other declarative notation, a search space for the problem, so that the robot could solve it with the General Graph Search program.

[10]

d) Sketch the search space of Part (c) for depth $d = 3$.

[2]

e) Justify, with reference to features of the search space shown in Part (d), which of the algorithms of Part (a) that the robot should choose to solve the problem.

[2]

2 a) Explain the operation of Uniform Cost, Best First, and the A* graph search algorithms.

[3]

b) Compare and contrast the performance of the three algorithms with respect to appropriate criteria.

[3]

c) In the context of A* search, explain why A* search is provably optimal amongst graph search algorithms of this type.

[5]

d) Given the (implicit) definition of a graph $G' = < S, Op >$, where S is a node and Op is a set of operators, explain how G' defines the same set of paths as the (explicit) definition of a graph $G = < N, E, R >$, where N is the set of nodes, E is the set of edges, and R is the incidence relation.

[6]

e) Explain how the A* search algorithm explores the paths defined by G'.

[3]

3 a) Explain, with an example, how the Alphabeta algorithm for 2-player games prunes branches of the search tree.

[6]

b) A group of four humanoid robots are travelling single-file on a very narrow ledge, and encounter a group of four other humanoid robots coming the other way.

As everyone knows, humanoid robots don't have a reverse gear, especially when on a precarious ledge. They can however climb over each other, but only if there is a robot-sized space on the other side.

As everyone also knows, humanoid robots don't have great visual sensors either, so they don't see each other until there is only exactly one robot-size space between the leading robots in the two groups.

The robots do however manage to negotiate that the groups will take it in turns to move. On each group's turn, one robot can move into an empty space, if it is adjacent to it; or a robot can climb over any other robot, if that robot is adjacent to an empty space; or the group may agree to pass their turn.

Formulate the problem space as a 2-player game for the General Graph Search program.

[10]

c) Briefly comment on the assumptions underlying game-playing search algorithms for 2-player adversarial games and co-operative games of the kind seen in Part (b).

[4]

4 a) Define the resolution inference rule.

[3]

b) Define a unification algorithm, and explain why it is important in resolution.

[3]

c) In *Bill, The Galactic Hero*, Bill is accused of being AWOL from the space troopers. The presiding judge makes the following statements:

If anyone is in the space troopers, they must be on duty.
If anyone has been on planet Helior for a year, they must have slept.
If anyone is on duty and has slept, they must have slept on duty.
If anyone sleeps on duty, they must be guilty.

Express these four statements as formulas of First Order Predicate Logic, and translate them into clausal form.

[4]

d) The judge accepts the following facts:

Bill is in the space troopers.
Bill has been on planet Helior for a year.

Express these facts in clausal form.

[2]

e) Prove, using resolution and showing the unifiers, that *Bill is guilty*.

[4]

f) Explain the relationship between Prolog's search strategy for a solution to a query and algorithms for graph search.

[4]

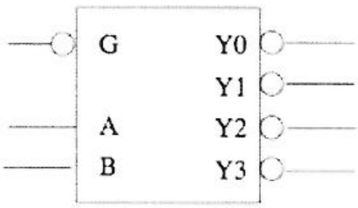
5 a) Specify the α and β elimination rules of the proof procedure KE.

[3]

b) Hence, or otherwise, specify a set of elimination rules for the equivalence operator $\leftrightarrow$.

[3]

c) Figure 5.1 shows a block diagram and truth table for a 2-input decoder. It works by interpreting a 2-bit binary input to have values 0..3. The input specifies which active low output from $\overline{Y0}$ to $\overline{Y3}$ will be active (false). If the active low enable $\overline{G}$ is inactive (true), then all the $\overline{Y[0..3]}$ outputs are inactive (true).



$\overline{G}$	B	A	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$
T	x	x	T	T	T	T
F	F	F	F	T	T	T
F	F	T	T	F	T	T
F	T	F	T	T	F	T
F	T	T	T	T	T	F

Figure 5.1: Block diagram for 2-input Decoder with Enable

2-input decoders can be used to implement any 2-input Boolean function by combining its output appropriately. The design in Figure 5.2 combines its $\overline{Y0}$ and $\overline{Y1}$ outputs using a nand-gate.

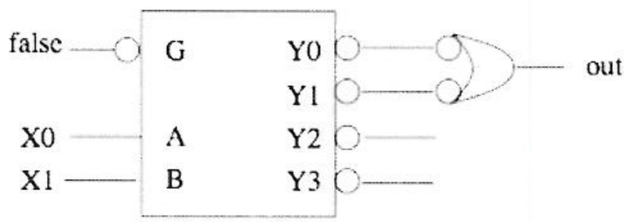


Figure 5.2: Decoder design for Boolean function

Prove, using the proof procedure KE, that $out \leftrightarrow \neg X1$.

It is essential that you annotate the KE-tree to show the inference steps.

[14]

6 a) Define a syntax for well-formed formulas of propositional modal logic. Define the semantics of well-formed formulas of propositional modal logic.

Explain why the formula $\Box p$ is true, and the formula $\Diamond p$ is false, at world w in model M when there is no world w' such that wRw' is in the accessibility relation of the model.

[6]

b) Consider the following frame conditions on the class of all models. State the corresponding axiom schema.

(i) *Reflexivity*.

(ii) *Symmetry*.

(iii) *Transitivity*.

(iv) *Seriality*.

[4]

c) Choosing *one* of the frame conditions defined in Part (b), show that the corresponding axiom schema holds in the class of models in which the accessibility relation satisfies the chosen frame condition.

Show that the same axiom schema does not hold in the class of all models.

[4]

d) Prove, using the KE calculus for propositional modal logic, that the axiom schema $\Box(p \rightarrow q) \rightarrow (\Box p \rightarrow \Box q)$ holds in both modal logics K and S5.

Explain why, using the KE calculus for propositional modal logic, that the axiom schema $\Diamond p \rightarrow \Box\Diamond p$, can be proved in modal logic S5 but not in modal logic K.

[6]

EE3-16

2012

The Answer

1 a) Bookwork.

Depth first: choose first path, work out one step extensions, append other paths to new paths.

Breadth first: choose first path, work out one step extensions, append new paths to other paths.

ID Depth first: Set depth = 0. Choose first path, if length = depth discard, otherwise work out one step extensions, append other paths to new paths. If no solution found at depth, add 1 to depth, and search at new depth.

[3]

b) Bookwork.

Depth first: exponential time, linear space, not complete not optimal

Breadth first: exponential time, exponential space, complete optimal (provided …)

ID DF: time space complexicty of BF; complete optimality of BF

[3]

c) Application

state representation

(location, direction, distance) of type integer, +/-, integer

Start state

(0, -, 0)

Goal state

(N, _, _) :- door_at(N') and |N| > |N'|

State change

Change direction

(0, Direction, Distance), (0, OppDirn, Distance) :- opposite(Direction, OppDirn)

Search

(0, Direction, Distance), (Location, Direction, Distance) :- Location is 2 ^ Distance

Return

(Location, Direction, Distance), (0, Direction, Distance)

This will cause the robot to search 1 unit +, 2 units -, 4 units +, 8 units -, etc

[10]

d) Application

Depends on the answer to c

[2]

e) Application

Depends on answer to d. This answer to c has loops so use breath first.

[2]

2 a) Bookwork

UC: pick path with frontier node with lowest actual cost given by path cost function g
BF: pick path with frontier node with lowest estimated cost given by heuristic h
A*: pick path with frontier node with lowest estimated cost of path through node to goal given by $g + h$

[3]

b) Bookwork

UC: time space exponential, optimal complete
BF: time space exponential but reduced substantially with good heuristic, not optimal not complete
A*: still looking at exponential complexity but reduced substantially with good heuristic and optimal and complete

[3]

c) Understanding.

Optimality justification:
Optimal solution has cost f^* to get to optimal goal G
Suppose A* search returns path to sub-optimal goal G'
We show that this is impossible

$f(G')$	$=$	$g(G') + h(G')$	
	$=$	$g(G') + 0$	G' is a goal state, we require h to be 0
	$=$	$g(G')$	

If G' is sub-optimal then $g(G') > f^*$
Now consider a node n on path to optimal solution G

```
    |-          n0 (start)
    |            /  \
    |           /    \
 f* |          n      G'
    |         /
    |        /
    |-     G
```

Then:	f^*	$\geq$	$f(n)$	monotonicity
	$f(n)$	$\geq$	$f(G')$	otherwise A* expands n first
	f^*	$\geq$	$f(G')$	transitivity of $\geq$
	f^*	$\geq$	$g(G')$	a contradiction

So either G' was optimal or A* does not return a sub-optimal solution.

Provably optimal: got to expand all the nodes with an f-cost $< f^*$ for a given h otherwise risk missing the optimal path

[5]

d) Understanding

Paths(G) = $_{i=0}\bigcup^{\infty}$ Pi
Where
P0 = { (<s>,0) }

Pi+1 = {(p ++ < n_{i+1}>, (cost+e+h(n_{i+1}))) | !(p_i,cost) □ Pi. frontier(p_i) = (n_i) & (n_i,e,n_{i+1}) □ R }

Since P'0 = P0, then every P'i+1 = Pi+1, on

Then the inductive definition of the paths and their costs in the graph is given by:

$P'_G = {}_{i=0}\bigcup^{\infty} P'_i$
where

$P'_0 = \{ (\langle s \rangle, 0) \}$

$P'_{i+1} = \{ (p_i \mathbin{++} \langle n_{i+1} \rangle, (cost+e+h(n_{i+1}))) \mid \exists\, op \in Op \,.\, \exists\, (p_i, cost) \in P'_i \,.\, (n_{i+1}, e) = op(frontier(p_i)) \}$

Clearly P0 = P'0. Then P'i = Pi for all i because frontier(p_i) = ni and op(ni) = (n_{i+1},e) if and only if (n_i,e,n_{i+1}) □ R

[6]

e) Understanding
Pick the path in Pi with the lowest cost+e+h(n)

[3]

3 a) Bookwork

◊Associate one of two values with each node

—Alpha value, associated with MAX nodes, which can never decrease
•Alpha is the 'least' MAX can get, given MIN will do its best to minimise MAX's value
-- Beta value, associated with MIN nodes, which can never increase
• Beta is the 'most' MAX can get, given MIN will do its best to minimise MAX's value

©**Algorithm**

◊Search to full ply using depth first

◊Apply heuristic evaluation to all siblings at ply

—Assume these are MIN nodes

◊Propagate value of siblings to parent using Minimax rules

—If MIN nodes, back up the maximum value

◊Offer this value to ***grandparent*** MIN node as possible beta cutoff

◊Descend to other grandchilren

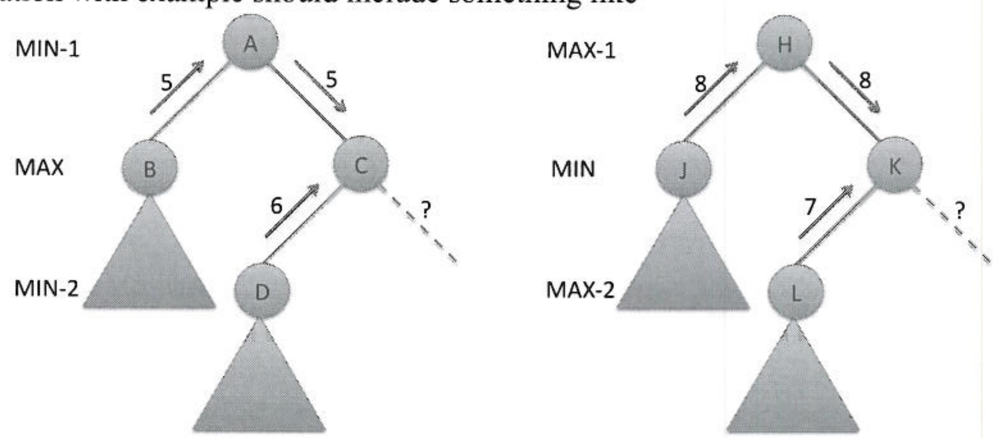
◊Terminate (prune) exploration of parent if any of their values is greater than or equal to the beta cutoff

◊Do the same for MAX nodes

◊Two rules for terminating search

—Search stopped below any MIN node having a beta value
less than or equal to alpha value of any of its MAX ancestors
Search stopped below any MAX node having an alpha value
greater than or equal to beta value of any of its MIN ancestors

Explanation with example should include something like



B returns 5 as its candidate MIN-1 value
A offer 5 to C as its beta-cutoff
D returns 6 as its candidate MAX value
at Node C, alpha value 5 > beta cut-off
so 6 will lose to 5 going from Max -> Min-1 level
any branch on the rhs under ?: if it returns < 6, it will lose at min-2 -> max level
if it returns > 6, it will (also) lose at max -> min-1 level

J returns 8 as its candidate MAX-1 value
H offer 8 to K as its alpha-cutoff
L returns 7 as its candidate MIN value
at Node K, beta value 7 < alpha cut-off
so 7 will lose to 8 going from Min -> Max-1 level
any branch on the rhs under ?: if it returns > 7, it will lose at max-2 -> min level
if it returns < 7, it will (also) lose at min -> max-1 level

[6]

b) Application

```
move
(p1,S), (p2,S') :-
append( Fr, [r1, space | Back], S ),
append( Fr, [space, r1 | Back], S'

climb
(p1,S), (p2,S') :-
    append( Fr, [r1, AnyRobot, space | Back] ),
    appendi Fr, [space, AnyRobot, r1 | Back], S' )

pass
(p1,S), (p2,S)
```

[10]

c) Application

Adversarial
Same and complete information
No communication

[4]

4 a) Bookwork
p + q, -p + r, infer q + r.

[3]

b) Bookwork.
unification of terms s and t is u (if it exists) which is a term that is a substitution instance of both s and t (ie a term that results from a consistent substitution of constants for variables)

[1]

algorithm
two constant unify if the are the same constant
a variable unifies with a term
two compound terms unify if
    they have the same functor
    they have the same arity (no arguments)
    the arguments piecewise unify

[3]

c) Application

```
@x.  troopers(x)  >  onduty(x)
@x.  helior(x)  >  slept(x)
@x.  onduty(x)  &  slept(x)  >  sleptonduty(x)
@x.  sleptonduty(x)  >  guilty(x).

-troopers(x)  +  onduty(x)
-helior(x)  +  slept(x)
-onduty(x)  +  -slept(x)  +  sleptonduty(x)
-sleptonduty(x)  +  guilty(x).
```

[4]

d) Application:

```
troopers(bill)
helior(bill)
```

[2]

e) Application

```
-guilty(bill)
-sleptonduty(bill)
-onduty(bill)  +  -slept(bill)
-troopers(bill)  +  -slept(bill)
-slept(bill)
-helior(bill)
contradiction
```

[4]

f) Understanding.

Left to right selection of goal, top to bottom selection of clauses is depth first search
Fast, even if incomplete, and not always "logic"

[4]

5 a) Bookwork

```
p & q
p

p + q, -p
q

etc
```

[3]

b) Understanding p < q == p>q & q>p

```
p < q, p
q

-(p < q), p
-q

etc
```

[3]

c) Application.

[14]

```
prem1       Y0 < -(-G & -X1 & -X0)              1
prem2       Y1 < -(-G & -X1 & X0)               2
prem3       out < -Y0 + -Y1                 3
prem4       -G                              4
neg conc    -(out < -X1)                        5

PB1/neg conc        out                         6
<,5,6           --X1                        7
--              X1                          8
>,3,6           -Y0 + -Y1                   9

                PB1.1/9    Y0               10
                +,9,10     -Y1              11
                <,2,11-G & -X1 & X0     12
§               &,12       -G               13
                           -X1              14
                           X0               15
                           Close 8 14
```

PB1.2/9	-Y0		16	
<,1,16	-G & -X1 & -X0	17		
&,12	-G		18	
	-X1		19	
	-X0		20	
	Close 8 19			

PB2/neg conc	-out			21
<,5,21	-X1	22		
<,3,21	-(-Y0 + -Y1)		23	
-+,23	Y0		24	
-+,23	Y1		25	
<,24,1	-(-G & -X1 & -X0)		26	
<,25,2	-(-G & -X1 & X0)		27	
-&,26,4	-(-X1 & -X0)			28
-&,27,4	-(-X1 & X0)			29
-&,26,4	-(-X0)		30	
-&,27,4	-(X0)		31	
	close 30 31			

6 a) Bookwork

wff ::= # wff |, $ wff

Kripke model M

M = <W, R, ||>

Where W is non-empty set of worlds

R is accessibility relation on W

|| is denotation function which maps propositions onto subsets of W

Meaning of modal formulas

⊨M,a#p is true < @w . aRw > ⊨M,wp

⊨M,a$p is true < !w . aRw & ⊨M,wp

With box p there is no world arW then antecedent is false so whole formula is true, but with diamond p left conjunct isfalse so whole formula is false

[6]

b) Bookwork/understanding

reflexive @w wRw

symmetric @ab aRb > bRa

transitive @abc aRb ^ bRc > aRc

serial @w !x wRx

[4]

c) Bookwork/understanding

Pick B

(i)

assume p, show box dia p

suppose p is true in some a

suppose a R b, any b

then b R a by symmetry

so dia p at b

since dia p is true at any (every) b accessible from a

then box dia p is true at a, as required

(ii)

M = <W,R,P>

W = {a,b}

R = {aRb}

||p|| = {a}

p is true in a

by MP box dia p true

so dia p true in b

but dia p false in b

[4]

d) Application

In K

[1] -(£(p > q) > (£ p > £q))

[1] £(p > q)

[1] -(£ p > £q)

[1] £p

[1]   -£q
[1,1]   -q
[1,1]   p>q
[1,1]   q
close

In S5
1   -(£(p > q) > (£ p > £q))
1   £(p > q)
1   -(£ p > £q)
1   £p
1   -£q
2   -q
2   p>q
2   q
close

In S5
1   -($p > £$p)
1   $p
1   -£$p
2   -$p
3   p
3   -p
close

In K
[1]   -($p > £$p)
[1] $p
[1]   -£$p
[1,1]   -$p
[1,2]   p
[1,1,1]-p

Because in S5 we can go to any world to any other, we can go from 2 to 3. But from [1,1] in K, [1,2] is only world available, but it is not accessible

[6]