

THE UNIVERSITY OF HONG KONG

FACULTY OF ENGINEERING  
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

CSIS0270 Artificial Intelligence

Date: May 30, 2003

Time: 2:00–5:00 pm

Candidates may use any calculator which fulfils the following criteria: (a) it should be self-contained, silent, battery-operated and pocket-sized; (b) it should have numeral-display facilities only and should be used only for the purpose of calculation; (c) it should not have any printing device, alphanumeric keyboard, or graphic display; and (d) it should not contain any recorded data or program. It is the candidate's responsibility to ensure that the calculator operates satisfactorily and the candidate must record the name and type of the calculator on the front page of the examination scripts. Lists of permitted/prohibited calculators will not be made available to candidates for reference, and the onus will be on the candidate to ensure that the calculator used will not be in violation of the criteria listed above.

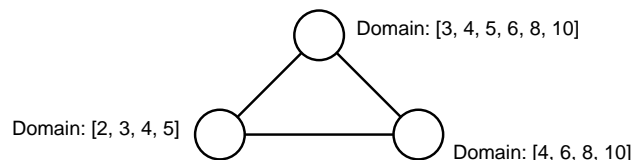
Answer all questions.

1. (Uninformed search)

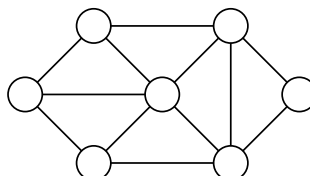
- a. (6%) Write the pseudocode of the IDS (iterative deepening search) algorithm, modified to allow the search to terminate if no node of the search tree is deeper than  $D_{\max}$ , although  $D_{\max}$  is unknown to the algorithm.
- b. (6%) Assume we are searching in a general (non-tree) state-space which is finite. Out of the 3 ways to avoid repeated states, which allow the above modified IDS to guarantee termination whenever there is no solution?
- c. (3%) Give one example where IDS is significantly better than BDS (bidirectional search), even though the problem is one which BDS can naturally be implemented.

2. (CSP)

- a. (6%) Perform MAC (Maintaining Arc Consistency) on the following graph, with each edge being a constraint "one of the number doubles or triples the other". Show whether the CSP has a solution.



- b. (6%) Give the minimum width tree decomposition of the following graph. What is its tree-width?



3. (Two player games)

- a. (6%) John has developed a program that plays a two-player deterministic game. He is worried about players trying all possible moves against his program to find a way to beat it. Therefore, he wants to add a probabilistic element into his program. Basically, once he obtains the utilities (which are positive integers) of each move, he wants his program to select a move with probability proportional to the utilities. He modifies the top-level Alpha-Beta, which now have the following structure.

```
def AlphaBetaDecision(state):
    alpha = 0
    for i in range(0, NUM_MOVES):
        valueof[i] = MinValue(Apply(state, i), alpha, infinity)
        if (valueof[i] > alpha):
            alpha = valueof[i]
    # select a move i based on valueof[i]
    # in the probability defined above
    return select_with_probability(valueof)
```

When John tests his program, he finds that the bad moves are chosen as likely as the good moves! Explain why his program has such a behaviour.

- b. (6%) Advice John how to fix the problem. Will it degrade the speed of the program? If so, suggest a way to counteract the speed degradation.

4. (Propositional Calculus Inference)

- a. (8%) Use DPLL to show whether there is a solution to the following satisfiability problem. When none of the heuristic is applicable, try variables in alphabetic order.

$$\begin{aligned} & A \vee B \vee \neg D \\ & A \vee \neg C \\ & A \vee C \\ & \neg A \vee \neg B \\ & \neg A \vee C \vee \neg D \\ & \neg A \vee D \\ & B \vee \neg C \end{aligned}$$

- b. (4%) Can you solve the problem using chaining? Explain your reasons.

5. (FOL and Resolution)

- a. (9%) The following sentences are about a relation  $R$ . Use FOL (First Order Logic) to represent them, and convert the sentences you write to CNF.

1. For any object  $x$ , there is an object  $y$  such that  $R(x, y)$  holds.
2.  $R$  is symmetric. (i.e., If  $x$  relates to  $y$ , then  $y$  also relates to  $x$ .)
3.  $R$  is transitive. (i.e., If  $x$  relates to  $y$ , and  $y$  relates to  $z$ , then  $x$  relates to  $z$ .)

- b. (9%) By using FOL refutation with linear resolution and Breadth First Search, prove that  $R$  is reflexive (for any object  $x$ ,  $x$  relates with itself), given all the above properties. Show all steps.

6. (Prolog)

- a. (8%) Show the AND-OR tree for the Prolog query  $a([p, x, q])$ , given the following clauses. Hence give all the answers to the query.

$a([X|Y]) :- b(Y), c(X).$   
 $b([]).$   
 $b([X|Y]) :- c(X), !, b(Y).$   
 $c(p).$   
 $c(q).$

- b. (8%) Without using any standard Prolog predicate, define  $my\_reverse(X, Y)$ , which is true when  $Y$  is the reverse list of  $X$ . E.g.,  $my\_reverse([a, b, c], [c, b, a])$  and  $my\_reverse([[1,2], a, 7], [7, a, [1,2]])$  should be true. It should run in linear time. You may add auxillary predicates for use with  $my\_reverse$  (Hint: write an auxillary predicate with an extra argument to accumulate elements you found in the head of one list). A slower implementation will score partial credit.

7. (Planning) The following planning problem is described in ADL (Abstract Description Language), where open-world assumption holds:

- Initial state:  $P1(A), \neg P2(A, B)$
- Goal:  $P2(A, B) \wedge P3(B)$
- Action 1:  $A1(x)$ , effects  $P1(x) \wedge$  (**when**  $P1(A) : P2(x, A)$ )
- Action 2:  $A2(x)$ , precondition  $P1(x)$ , effects  $P3(x)$
- Action 3:  $A3(x)$ , precondition  $P3(x) \wedge P2(B, x)$ , effects  $P2(x, B) \wedge \neg P3(B)$

- a. (6%) Convert the planning problem to STRIPS representation, where the close-world assumption holds, preconditions cannot have negated predicates, and effects cannot be conditional. (Hint: you may need to introduce your own predicates and actions.)
- b. (6%) Give a smallest (i.e., fewest actions and ordering constraints) partial-order plan that solves the original problem (in ADL format). Show all the causal links with arrows, and ordering constraints with dashed arrows. You may omit ordering constraints that is implied by transitive properties or by causal links.
- c. (3%) Give all linearizations of the partial-order plan you give in part (b).

**END OF PAPER**