DHARMSINH DESAI UNIVERSITY, NADIAD
FACULTY OF TECHNOLOGY
ONLINE SESSIONAL EXAMINATION

B.Tech (CE) sem → 7
SUBJECT → Artificial Intelligence

Roll No → 142
Signature → Prakruti

Date : 24/8/2020
Time : 10:00 to 11:15
Total Pages : 10

Q. 1

A.

→ This is an example of left recursion as it uses its own execution.
→ It is useful when we need to perform 'something' repeatedly.
→ But to stop executing it somewhere, we also need to declare some facts also.

B.

Cut :→ (C denoted by !.)

→ it is used for explicitly declaring the result as true always, so that it doesnt go for alternatives (i.e. alternative predicates)
→ also for stoping Back trace and exploring other options.

fail :-> it is a predicate without any argument.

→ it is used for explicitly stating the predicate as fail (false)

→ so that, we can go through all the alternative solutions i.e. branches of the tree.

C. And elimination

Premise $A \wedge B$ (i.e. if $A \wedge B$ is true A is true)
conclusion A

and a rule is sound if conclusion is true whenever premise is true.

→ & And in above rule conclusion A will be true if $A \wedge B$ is given true. &
It can be checked through truth table.

→ So it is sound.

D. To pass the turing test, a ~~mute~~ machine should,

→ Natural language processing

→ To interact with human understandable statements and behave like a human.

→ Machine learning.

→ to acquire intelligence from given knowledge & rules &

improve the performance.

→ knowledge representation

→ to present knowledge in application in a proper way.

→ Reasoning.

→ to choose appropriate way to reach to solution.

E.

F. The main difference is the way in which both methods update the current solution.

→ Simulated Annealing can escape from local optimum accepting worst solutions with a probability.

→ hill climbing only updates when it finds a better solution, following the gradient of exact function

Q. 2

A. Length of input list

domains
list = integer *
Predicate
length (list, integer)

clauses
length ([ ], 0).
length ([_ | Tail], N1) :-
length (Tail , N2),
N1 = N2 + 1.

execution trace → length ([2, 1, 6], X)

call: length ([2, 1, 6], X) (2nd pred, as its not empty)
call : length ([1, 6], _) (2nd pred)
call : length ([6], _) (2nd pred)
call : length ([ ], _) (1st pred)
return : length ([ ], 0)
N1 = 0 + 1
return : length ([6], 1)
(N1 = 1 + 1)
return : length ([1, 6], 2)
(N1 = 2 + 1)
return : length ([2, 1, 6], 3)

Output → X = 3
1 solution.

B. sum of N natural numbers recursively.

Predicate
sum (integer, integer)

clauses

```
sum(0,0).
sum(N, S1) :- N>0,
              M = N-1,
              sum(M, S2),
              S1 = S2+N.
```

execution trace → sum(5, X)

call: sum(5, X)
(as 5>0, M=4)
call: sum(4, X)
(as 4>0, M=3)
call: sum(3, X)
(as 3>0, M=2)
call: sum(2, X)
(as 2>0, M=1)
call: sum(1, X)
(as 1>0, M=0)
call: sum(0, _)
return: sum(0,0)
($S_1 = 0+1$)
return: sum(1,1)
($S_1 = 1+2$)

(6)

return: sum(2, 3)
($S_1 = 3+3$)
return: sum(3, 6)
($S_1 = 6+4$)
return: sum(4, 10)
($S_1 = 10+5$)
return: sum(5, 15)
X

output → X = 15
1 solution.

Q. 3

A.

(a)(i) can solution steps be ignored or undone?

Travelling salesman: → ~~same~~ ~~is necessary to since the~~ recoverable
we can't back back

Cannibals: → irrecoverable
as we lose the game if we make wrong move.

(ii) is universe predictable?

Travelling salesman → uncertain outcome

- as there may be multiple paths to cover having minimum distance travelling, so we may get different solutions

Cannibals: → final state is precise. so, certain outcome.

(iii) good solution absolute or relative?

Travelling salesman → it can be considered as any-path problem since we don't to use heuristics

& get a good solution in less computation.

cannibals :- as there can be multiple ways to solution, we may define some heuristics that help us reaching final state in less steps.
→ so it can be considered as any-path problem.

(iv) state or path?

Travelling Salesmen :→ as we need the path to follow, not only that we succeded the task.
→ so, to get path, it should be a path solution.

cannibals :→ as we have to specify how we solved the problem, it is a path solution.

(b) issues in design of search programs.

→ as search programs can be seen as trees, we face below problems.

- Perform search in forward direction or backward direction?
- how to pick rules that are applied so that we have efficient procedure to reach to solution.

- how to show nodes in tree?
- what if some node is generated twice?
- what if we reach to starting node again?

B.

(a) disadvantages of Hill climbing.

→ decides further steps by looking only at immidiate points to the current state.

→ so if, current state is a local optimum then we tend to miss the best solution.

→ Ridges cause difficulties for it

→ flat regions cause algorithm having no directions to move further

solutions :→

→ Take big steps or very little steps while searching according to the problems in case of flat region.

→ Backtrack to solve local optimum problem. Try different path that seem to be promising.

→ incase of ridges, try moving in different directions so that we can have best solution.

(b) let initial position is p_1 & we need to go to position p_2

& the time it takes is t

so we can define state as

(position, time remaining)

and we can choose the operations which cause the minimum time to reach the place

—— end of answer sheet ——