TI1600-A Multi-Agent Systemen

18 April 2012

This exam will test your knowledge and understanding of the material provided to you and presented in the lectures, the book of Blackburn, Bos and Striegnitz *Learn Prolog Now!* (Chapters 1 to 5, Section 10.3, Section 11.2), as well as *The* GOAL *Programming Guide* and the paper "Common ground and coordination in joint activity" (Klein et al.). It is *not* allowed to use materials such as books, papers or slides during the exam. You will have 3 hours (from 9 till 12) to complete the exam. You may provide your answers in Dutch as well as in English. It has 7 questions, for a total of 100 points. Please don't include irrelevant information: you will be marked down for this. Before you hand in your answers, please check that you have put your name and student number on top of every sheet you hand in. Answers may be provided in Dutch.

Questions

Question 1 *10 points*

This assignment concerns Prolog.

(a) (10 points) Which of the following queries succeed? If a query succeeds, provide the unifier that Prolog computes (if it is non-empty). For example, the query `f(X) = f(a).` succeeds, and Prolog computes the unifier `X=a`; the query `5 = 5` succeeds, and the unifier is empty (i.e., Prolog returns `true`).

```
1) p(X,f(a,b),g(Y)) = p([], f(Z,Y), g(b)).
2) [1,2,3,[]|[]] = [1,2,3,[]].
3) X is 2+3, f(X) = f(5).
4) 2+3 is 1+4.
5) [f(3), a, [], [3]] = [_,X,Y|Z].
```

Question 2 *15 points*

This assignment concerns Prolog.

(a) (7 points) Write a (set of) clause(s) that define the predicate `nth(N,L,X).` where `N` is an integer greater than 0 and `L` is a non-empty list, such that `X` is the `N`-th element of `L`. For example, the query `nth(3,[1,2,5,8],X).` should yield `X = 5`. *Do not use any auxiliary (=additional) predicates!*

(b) (8 points) Write a (set of) clause(s) that define the predicate `minus(L1,L2,L3)` where `L1`, `L2` and `L3` are non-empty lists of integers, and `L3` is obtained from `L1` and `L2` by subtracting the i-th element of `L2` from the i-th element of `L1`, where i ranges from 1 to the length of the lists. One could say that we compute "`L1` minus `L2`". The query should fail if `L1` and `L2` are not of equal length. For example, the query `minus([3,4,5],[1,2,3],L).` should yield `L = [2, 2, 2].`, and `minus([3,-4,5],[5,-2,3],L).` should yield `L = [-2, -2, 2]`. *Do not use any auxiliary (=additional) predicates!*

Question 3 *5 points*

This assignment concerns Prolog. Consider the following program:

```
a(5).
b(1).
f(3).

g(1).
g(2).

p(2).
p(3).
p(X) :- a(X).
p(X) :- b(X).

px([]).
px([X|T]) :- g(_), p(X), not(f(X)), px(T).
```

When posing the query `px([1,3]).` to Prolog, the following trace is produced (only the first part is displayed here):

```
[trace]  ?- px([1,3]).
   Call: (7) px([1, 3]) ? creep
   Call: (8) g(_G454) ? creep
   Exit: (8) g(1) ? creep
```

```
   Call: (8) p(1) ? creep
   Call: (9) a(1) ? creep
   Fail: (9) a(1) ? creep
   Redo: (8) p(1) ? creep
   Call: (9) b(1) ? creep
   Exit: (9) b(1) ? creep
   Exit: (8) p(1) ? creep
^  Call: (8) not(f(1)) ? creep
   Call: (9) f(1) ? creep
   Fail: (9) f(1) ? creep
^  Exit: (8) not(user:f(1)) ? creep
   Call: (8) px([3]) ? creep
....
```

(a) (5 points) In the text below that explains the trace, choose from the following terms to complete the text. Terms should be inserted in the appropriate slots, where each term can be used zero or more times:

- backtracking
- recursion
- breadth-first search
- depth-first search
- linear search
- negation as failure

Prolog unifies `px([1,3]).` with the head of the last clause in the program. Prolog applies `__________` (1) and thus tries to prove `g(_G454)` before trying to prove `p(1)`. Prolog applies `__________` (2) and thus unifies `g(_G454)` with `g(1)` rather than with `g(2)`. The goal `p(1)` is unified with the head of the first rule in the program, and Prolog tries to prove `a(1)`. This fails and Prolog applies `__________` (3), which results in the application of the second rule in the program, and Prolog proves `b(1)`, thereby proving `p(1)`. Then Prolog tries to prove `not(f(1))`, which succeeds because Prolog uses `__________` (4) which means that `not(f(1))` succeeds if `f(1)` fails. Prolog applies `__________` (5) and tries to prove `px([3])`.

Question 4 *6 points*

(a) (3 points) Mention an application where you think agents can be applied naturally. Explain why you think so (use 5 sentences at most).

(b) (3 points) A software agent is usually connected to an environment and can perceive things in its environment. Most environments, however, are not fully observable. Discuss how an agent can try to obtain a complete representation of its environment (use 5 sentences at most).

Question 5 *13 points*

This question concerns the agent programming language GOAL.

(a) (2 points) List the mental state components of a GOAL agent. Which of these components can be modified at runtime?

(b) (4 points) Explain what a mental state condition is and how a condition of the form `bel(`*Query*`)` such as `bel(clear(X))` is evaluated.

(c) (4 points) Explain the difference between rules of the form `if...then` and rules of the form `forall...do`.

(d) (3 points) In which module in an agent program would you typically use a rule of the form `forall...do`? Explain why.

Question 6 *41 points*

This question concerns the agent programming language GOAL. Consider the agent program below. This program is written for a robot that collects rocks on planet Mars. Some rocks are interesting to take home but others are not. The robot can maximally carry 5 rocks and then has to return to its home base to deliver the collected rocks.

(a) (10 points) Explain which action(s) the GOAL agent may perform *next*, given the agent program listed below. Only provide actions that it can perform given its "current", i.e. initial, mental state.

(b) (3 points) The `pickup` action does not take into account yet that the robot can carry maximally 5 rocks. Fix this issue and provide an action specification for `pickup` that correctly takes this constraint into account.

(c) (3 points) The agent does not take into account yet whether a rock is interesting or not to collect. Only yellow rocks are interesting. Modify the agent in a way that it will adopt goals to collect interesting rocks and only such goals. Explain why your answer implements this constraint.

(d) (10 points) Complete the action specification for the action `drop(Rock)` for dropping a rock `Rock`. A rock, of course, can only be dropped if it is being carried by the robot and after dropping the rock the robot should believe that the rock is located where the robot is. Only use the predicates that are already available in the given agent program. Motivate the pre- and postcondition that you have given.

(e) (5 points) The agent is not able to achieve its goals. Explain why.

(f) (10 points) Provide a modification of the agent program to ensure that the agent will achieve its goals. Argue why your answer solves the issue and motivate why you choose to modify the program in the way you did. Take into consideration in your motivation whether your solution is a minimal modification of the program.

```
init module{
  knowledge{
    interestingRock(Rock) :- color(Rock, yellow).
    itemsCollected(Items) :- setof(Item, carrying(Item), Items).
    nrOfitemsCollected(Count) :- itemsCollected(Items), length(Items,Count).
  }
  beliefs{
    home(a).
    at(rock1,b). at(rock2,a). at(rock3,c). at(rock4,c).
    color(rock1,yellow). color(rock2,blue). color(rock3,red). color(rock4,yellow).
    at(robot,a).
  }
  actionspec{
    pickup(Rock){
      pre{ at(robot,X), at(Rock,X) }
      post{ carrying(Rock), not(at(Rock,X)) }
    }
    goto(Y){
      pre{ at(robot,X) }
      post{ at(robot,Y), not(at(robot,X))}
    }
    drop(Rock){
      pre{ ... }
      post{ ... }
    }
  }
}
```

```
main module{
  program{
    if bel(at(R,X), not(R=robot), home(Y), not(X=Y)) then adopt(at(R,Y)).
    if goal(at(R,X)), bel(at(R,Y), at(robot,Z)) then goto(Y).
    if goal(at(R,X)), bel(at(R,Y), at(robot,Y)) then pickup(R).
    if bel(carrying(R), home(X)) then goto(X).
    if bel(carrying(R), home(X), at(robot,X)) then drop(R).
  }
}
```

Question 7 *10 points*

This question concerns joint activity in the BlocksWorld for Teams (BW4T) environment (see figure). BW4T is a virtual world containing nine rooms in which colored blocks are hidden, and a drop zone where blocks can be delivered. The BW4T task is to deliver a sequence of colored blocks in a particular order (displayed at the bottom of the figure). Agents in the BW4T world can move, pick up blocks, drop blocks and communicate with each other. An agent can only carry one block at a time. Agents know which rooms there are and which colors need to be delivered in what order, but they can only see blocks and their colors when they are inside the room where these blocks are. Agents cannot see each other. To deliver a block successfully, an agent has to find a block of the right color, go to the block, pick it up, go to the drop zone and drop the block there. Agents can communicate with each other to coordinate their actions. Performance on the BW4T task is measured by the speed of completing the task.

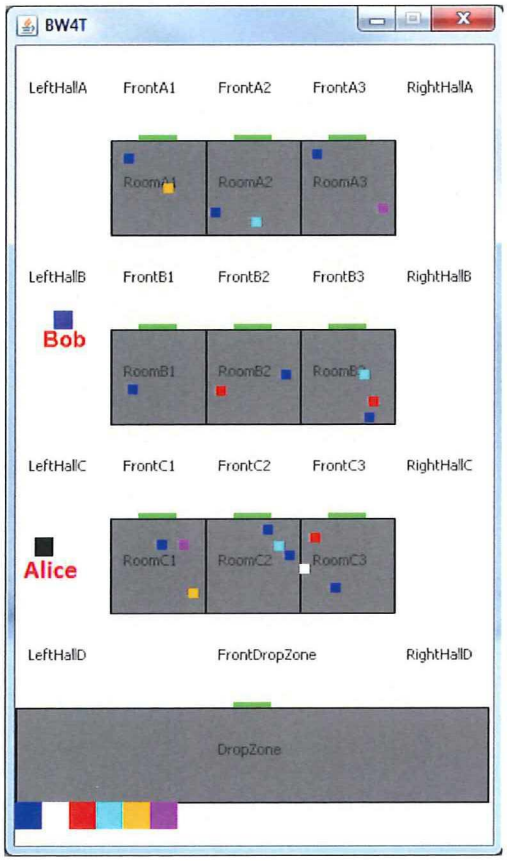

Figure 1: BW4T world

This question concerns four agents that want to perform the BW4T task. Each of the agents suffers from one or more handicaps that hinder task execution, and therefore they decide to form a team and perform the BW4T task together. The handicaps of the agents are the following.

- Agent 1 is not able to carry blocks and cannot remember the locations and colors of blocks
- Agent 2 and 3 are color blind and cannot remember the locations and colors of blocks
- Agent 4 is not able to move

(a) (2 points) Are the criteria for joint activity (intention and interdependence) satisfied in this team? Explain your answer.

(b) (5 points) Propose an organizational structure for this team by specifying one or more roles for each of the four agents, and motivate your choice. All four agents should contribute to the teams performance.

(c) (3 points) Propose three conventions regarding communication that would enhance the teams performance. In a communication convention you should specify the role of an agent, and when and what the agent should communicate.

End of exam