CIA2326: Coursework
Formal Aspects of Computer Science

Background and Context

In safety critical applications of software systems it is considered good practice to develop software rigorously. This often means adopting some kind of "formal method" to the software development process. One way of ensuring that requirements are captured accurately is to adopt an approach of trying to capture requirements formally within a model, using for example a logic language, and then animating and testing the requirements model before system development. In this coursework you are required to capture some requirements of a fictional air traffic control system in first order logic, and to animate that specification using the logic language Prolog.

In an air traffic control system managing an air space, such as the North East Atlantic, flight plans (called 'Flight Profiles') are logged with an Air Traffic Control Officer (ATCO) a period of time before the aircraft is due to enter that air space. At this time the system must check that all logged flight profiles are mutually conflict free.

Assume you are called in to create a set of requirements for a Flight Profile separation checking program (a 'conflict probe'). You learn that an aircraft's Flight Profile is a sequence of 2 or more segments points. You decide to represent the name of a Profile with the aircraft's callsign, eg ba202, and the segment make up of a Profile as a set of facts. For example, if ba202 is made up of 3 segment points, it can be recorded as follows:

belongs(ba202-1,ba202)
belongs(ba202-2,ba202)
belongs(ba202-3,ba202)

and another Flight Profile vgn902 is made up of 2 segment points, can be recorded as follows:

belongs(vgn902-1,vgn902)
belongs(vgn902-2,vgn902)

Further, each segment point is defined by

- a time (when the plane is due to fly over it), described as a tuple t(X,Y), meaning Y minutes past X o'clock on the 24 hour clock
- a flight level (fl), described as an integer X meaning 100*X feet above sea level
- a latitude (lat), described by an interger representing degrees north, latitude,
- a longitude (long), described by an interger representing degrees west, longitude
- a speed measured in Mach units, where Mach 1.0 is the speed of sound.

You decide to represent these as logical facts. For example, the time, flight level, latitude, longitude and speed of a segment point ba202-1 is be recorded as follows:

time(ba202-1, t(8,23))
fl(ba202-1,370)
long(ba202-1,21)
lat(ba202-1,65)
speed(ba202-1, 0.9)

meaning that flight ba202 is intending to be at 65 degrees north, 21 degress west (somewhere over Iceland) at 37000 feet at 23 minutes past 8 in the morning, and flying at Mach 0.9. As another example you might have a segment point of vgn902 represented as:

time(vgn902-1, t(8,23))
fl(vgn902-1,370)
long(vgn902-1,20)
lat(vgn902-1,65)
speed(vgn02-1, 0.85)

More examples of facts for Flight Profiles are given on http://scom.hud.ac.uk/scomtlm/cia2326/airspacefacts1 After consulting Air Traffic Control Officers and studying the rule books you gather the following requirements of a 'conflict probe' sub-system, concerning the logic of separation criteria, as follows:

1. Two Flight Profiles x and y are in Conflict if there exists a segment point xp of x and a segment point yp of y which are not separated.
2. Two segment points are Separated if they are Vertically Separated or Horizontally Separated or Time Seperated.
3. Two segment points are Vertically Separated if the Flight Level of one point is at least 2000 feet less than or greater than the flight level of the other point.
4. Two segment points are Horizontally Separated if the integer 2-D co-ordinate of one point - that is its (latitude,longitude) - is not equal to the 2-D co-ordinate of the other point.
5. Two segment points are Time Separated if the time of one point is at least X mins different from the time of another where X is the minimum time separation.
6. The minimum time separation (mts) between 2 segment points is at least: (a) 30 minutes when both of the aircraft are flying at supersonic speeds and at least one of the aircrafts' profiles are not in level flight (b) 20 minutes when both of the aircraft are flying at supersonic speeds and both of the aircrafts' profiles are in level flight (c) 10 minutes when one of the aircraft is flying subsonic and one is flying supersonic (d) 5 minutes if both aircraft are flying subsonic.
7. A Profile is not in level flight if there exists 2 segments belonging to the profile which have different flight levels.

Question 1

The definition of a Profile not being in level flight (criteria 7 above) can be represented in first order logic as follows:

$$\forall p(\\ (\exists s \exists t \exists l_s \exists l_t \ \ belongs(s,p) \wedge belongs(t,p) \wedge \\ fl(s,l_s) \wedge fl(t,l_t) \wedge \neg equal(l_s,l_t)) \\ \rightarrow notlevelflight(p))$$

(a) Systematically translate this wff into CLAUSAL FORM.
(b) Combining the clausal form of (a) with the facts about the Profile ba202 in file airspacefacts1, prove, using Resolution Refutation, that Profile ba202 is not in level flight. To do this, produce a proof tree annotated with the substitutions you need at each step. To complete your proof, you may also assume obvious facts about equality (for example, $\neg equal(380, 370)$).
(c) Part of an automated validation check for specifications can be to check if definitions are 'complete'. For example, if we had a definition of the form:

$$precondition1 \rightarrow q \\ precondition2 \rightarrow r$$

then this is equivalent to checking that 'precondition1 v precondition2' is valid. Consider the definition of mts in criterion 6: if it were possible to find an assignment of truth values that makes each of the four preconditions false, then in that case the system would fail to find a value for mts, potentially causing an insecurity. Using a truth table argument or otherwise, either show the definition in criterion 6 is complete or find an assignment of T/F for individual conditions that does not satisfy any of the preconditions.

Hand in: The clausal form of (a), the workings of translating the wff into clausal form, the Resolution Refutation tree showing substitutions at each resolution step for (b). For (c), hand in a textual argument augmented with truth table(s) if used.

Question 2

(a) Criteria 1,2,3 could be translated into Prolog clauses as follows:

```
conflict(P1,P2) :-
     belongs(S1,P1),
     belongs(S2,P2),
     \+ seperated(S1,S2).
seperated(S1,S2) :-
     vert_seperated(S1,S2).
seperated(S1,S2) :-
     horiz_seperated(S1,S2).
seperated(S1,S2) :-
     time_seperated(S1,S2).
vert_seperated(S1,S2) :-
     fl(S1,X), fl(S2, Y),
     ( X =< Y-20 ; X >= Y+20).
```

In this manner, and using the insight from Question 1, use all of criteria 1 - 7 and the facts about Profiles in file airspacefacts1 to create a Prolog program which captures the logic of this part of the separation standard. Try to write your program to reflect the logic of the requirements. Test out your Prolog program using the Profile examples given in the set of examples in file airspacefacts1. Make up another Profile of your own called "myplane", and use it in the tests.
(b) Now upgrade your program to perform the following function: it inputs a LIST of profile names, and outputs ALL pairs of profiles in the list that are NOT separated.

Hand in: Listings of your Prolog code. Screenshots for parts (a) and (b), showing tests (hard copy + electronic copy).

Question 3

(a) The file http://scom.hud.ac.uk/scomtlm/cia2326/atclogic contains some of the actual formalised separation criteria. Compare the actual criteria in this file, with the criteria given above in this coursework. Write a short report describing the differences. (b) The file http://scom.hud.ac.uk/scomtlm/cia2326/atcprolog contains Prolog code that has been automatically generated from the criteria. Comment on the challenges involved in automatically generating such a Prolog program from these criteria. You may consult the following publication for assistance (pages 20-22): http://scom.hud.ac.uk/scomtlm/Artform/pubs/spe_paper.pdf
Hand in: A short report of approximately 800 words addressing the two issues above (hard copy).

Marking Criteria

The coursework assesses outcomes 19.1.1 and 19.2.1-19.2.3 from your module specification. It makes up 40 per cent of your 20 credit module.

Approximate Marks Breakdown: 1: 30 per cent, 2: 45 per cent, 3: 25 per cent.

To calculate your mark the following **marking criteria** will be used:

- the correctness of the clausal form translation, proof tree and annotations, and arguments in Question 1,
- the accuracy and quality of the code with respect to the requirements, and the quality and correctness of the tests you perform.
- the insights, clarity, and cogency of your report in response to Question 3.

This coursework must be undertaken individually. The course work will be handed out during week 6, 2010. The deadline for handing in the work is Thursday 4pm December 16th 2010. The marked work will be handed back by the second lecture of Term 2.