Mid Year Exam
April 2008

Question1

[2 marks]

a) What are the main aspects of any ***knowledge representation language?***

A knowledge representation language is defined by two aspects:

- ***1. Syntax The syntax of a language defines which configurations of the components of the language constitute valid sentences.***
- ***2. Semantics The semantics defines which facts in the world the sentences refer to, and hence the statement about the world that each sentence makes.***

b) What is the difference between semantic net and frame system?

	semantic net	**frame system**
Representation	Concept as nodes, relationship as arcs	Frame consist of Frame identification Relationship to other frames Descriptors of the requirements Procedural information Default information
Declaration	Graphical way for representing declarative knowledge	Next development, after semantic network. A frame is viewed as concept in object oriented as it integrates declarative and procedural notation
inference mechanism	Two methods to do this: **Intersection search** Inheritance	Inheritance and in case of multiple inhertance cconstruct a class precedence list

Question2

[6 marks]

(1 mark)Transform axioms into clause form:

1- ∀ x (boy(x) v girl(x) -> child(x))

a- !(boy(x) v girl(x)) v child(x)
b- (!boy(x) ^ !girl(x)) v child(x)
c- (!boy(x) v child(x)) ^ (!girl(x) v child(x))

2- ∀ y (child(y) -> (gets(y,doll) v gets(y,train) v gets(y,coal)))

a- !child(y) v gets(y,doll) v gets(y,train) v gets(y,coal)

3- ∀ w (boy(w) -> !gets(w,doll))
!boy(w) v !gets(w,doll)

4 ∀ z (child(z) ^ good(z) -> !gets(z,coal))

a- !(child(z) ^ good(z)) v !gets(z,coal)
b- !child(z) v !good(z) v !gets(z,coal)

5 boy(Jack)

Construct a proof by refutation using resolution of the statement:
Negated conclusion (1 mark)

- !gets(Jack,train) -> !good(Jack)

!(!gets(Jack,train) -> !good(Jack))
!(gets(Jack,train) v !good(Jack))
!gets(Jack,train) ^ good(Jack)

4- The set of CNF clauses:

a- (a) !boy(x) v child(x)
(b) !girl(x) v child(x)
b- !child(y) v gets(y,doll) v gets(y,train) v gets(y,coal)
c- !boy(w) v !gets(w,doll)
d- !child(z) v !good(z) v !gets(z,coal)
e- boy(Jack)
f- (a) !gets(Jack,train)
(b) good(Jack)

5- Resolution: (1 mark)

- 4. !child(z) or !good(z) or !gets(z,coal)
6.(b). good(Jack)

----

7. !child(Jack) or !gets(Jack,coal) (substituting z by Jack)

- 1.(a). !boy(x) or child(x)
5. boy(Jack)

----

8. child(Jack) (substituting x by Jack)

- 7. !child(Jack) or !gets(Jack,coal)
8. child(Jack)

----

9. !gets(Jack,coal)

- 2. !child(y) or gets(y,doll) or gets(y,train) or gets(y,coal)
8. child(Jack)

----

10. gets(Jack,doll) or gets(Jack,train) or gets(Jack,coal) (substituting y by Jack)

```
o  9. !gets(Jack,coal)
   10. gets(Jack,doll) or gets(Jack,train) or gets(Jack,coal)
   ------------------------------------------------------
   11. gets(Jack,doll) or gets(Jack,train)
o  3. !boy(w) or !gets(w,doll)
   5. boy(Jack)
o  ------------------------------------------------------
   12. !gets(Jack,doll) (substituting w by Jack)
o  11. gets(Jack,doll) or gets(Jack,train)
   12. !gets(Jack,doll)
   ------------------------------------------------------
   13. gets(Jack,train)
o  6.(a). !gets(Jack,train)
   13. gets(Jack,train)
   ------------------------------------------------------
   14. empty clause
```

Consider the following set of rules that describe when a person can vote in a presidential election.

- **R1:** IF ?x was born in the Egypt THEN ?x is an Egyptian
- **R2:** IF ?x received Egyptian citizenship THEN ?x is an Egyptian
- **R3:** IF ?x's age >= 18 THEN ?x is an adult
- **R4:** IF ?x is Egyptian AND ?x is an adult THEN ?x can vote

Assume that the operator ">=" (greater than or equal) is a basic operator implemented in the inference engine.
The working memory contains the following assertions:

- **F1:** Ahmed's age is 16.
- **F2:** Amera received Egyptian citizenship.
- **F3:** Ahmed was born in the Egypt.
- **F4:** Amera's age is 20.

1-Use backward chaining to determine whether or not Ahmed can vote. Show the steps followed by backward chaining and how the working memory is updated (1.5 Mark)

```
                        Bill can vote
                              | (Using R4)
                              |
                      Bill is an American
                      / (Using R1)      \ (Using R2)
                     /                   \
       Bill was born in the US          Bill receives US
citizenship
          | (succeeds!)
          | WM <- {Bill is an American.}        Fails!
       Bill is an adult
          | (Using R3)
          |
       Bill's age >= 18

           Fails!

       So, Bill cannot vote!
```

2-(1.5 Mark)

3- Use forward chaining determine whether or not Amera can vote. Show the steps followed by forward chaining and how the working memory is updated

```
 Using R1
                         |       ?x was born in the US
                         |
                 Bill was born in the US

                      Succeeds!
                 WM <- { Bill is an American.}

    Using R2
                         |       ?x received US citizenship
                         |
                 Sue received US citizenship

                      Succeeds!
                 WM <- { Sue is an American.}

    Using R3
                         |       ?x's age >= 18
                         |
                 Sue's age >= 18

                      Succeeds!
                 WM <- { Sue is an adult.}

    Using R4
                                /\       ?x is an American
                            /        \
                         /              \
       Bill is an American.           Sue is an American
                |  ?x is an adult                   |  ?x is an adult
                |                             |
       Bill is an adult              Sue is an Adult

            Fails!                      Succeeds!
                                    WM <- { Sue can vote.}


```

Question3

[4 marks]

Consider the following Prolog program:

```
successor(2,1).
successor(3,2).
successor(4,3).
successor(5,4).
successor(6,5).
successor(7,6).
successor(8,7).
successor(9,8).

larger_digit(X,Y) :-
            successor(X,Y).
larger_digit(X,Z) :-
            successor(Y,Z),
            larger_digit(X,Y).
```

(a) What would be Prolog's answer to the query `successor(2,3)`. [1 mark] NO

(b) What would be Prolog's answer to the query `larger_digit(7,3)`. [1.5 marks] YES

(c) What would be Prolog's answer to the query `larger_digit(3,7)`. [1.5 marks] NO