York University- Department of Computer Science and Engineering

SC/CSE 3401 3.00 – Functional and Logic Programming

Solutions to assignment 1

1) (8 marks) Consider the following formulae:

$$A: \quad p \rightarrow (q \vee r)$$
$$B: \quad q \equiv r$$
$$C: \quad \neg q \wedge (p \vee r)$$

a) Using truth tables, which of the above formulae is satisfiable?
b) Which of the above formulae is a contradiction?
c) Which of the following sets are satisfiable?
   i. {A, B}
   ii. {B, C}
   iii. {A, B, C}
d) Which of the above sets i-iii are inconsistent?

Explain your answers.

Answer.

Showing state values true as 1 and false as 0, the truth tables for formula A, B, and C is:

	p	q	r	$q \vee r$	**A:** $p \rightarrow (q \vee r)$	**B:** $q \equiv r$	$\neg q$	$p \vee r$	**C:** $\neg q \wedge (p \vee r)$
1	**0**	**0**	**0**	**0**	1	1	1	**0**	**0**
2	**0**	**0**	1	1	1	**0**	1	1	1
3	**0**	1	**0**	1	1	**0**	**0**	**0**	**0**
4	**0**	1	1	1	1	1	**0**	1	**0**
5	1	**0**	**0**	**0**	**0**	1	1	1	1
6	1	**0**	1	1	1	**0**	1	1	1
7	1	1	**0**	1	1	**0**	**0**	1	**0**
8	1	1	1	1	1	1	**0**	1	**0**

a) All 3 formulae, A, B, and C, are satisfiable since there is at least one row which makes them true. For example row 1 shows a state which makes formula A true. Row 4 is a possible state that can make formula B true. Row 5 is a possible state than can make formula C true.
b) None of the 3 formula A, B, or C is a contradiction, since all of them are satisfiable.
c) For the sets:
   i. {A,B} is satisfiable. There exists a state (for example row 1) that makes both formulae true.
   ii. {B, C} is satisfiable. There exists a state (row 5) that makes both formulae true.
   iii. {A, B, C} is not satisfiable. There is no state that can make all three formulae true.

d) Sets {A,B} and {B,C} are NOT inconsistent, since they are satisfiable. On the other hand, set {A,B,C} IS inconsistent, since it is not satisfiable. For all possible states in the truth table, at least one of the formula A, B, or C is false (for example in row 1, C is false; in row 2, B is false; etc.).

---

2) (5 marks) Consider the following formula in domain of natural numbers:

$$(\forall x)(\forall y)(x > y \ \vee \ x+1=5)$$

In this formula,

(a) List all object variables and constants.

(b) What are the predicates? What are the functions?

(c) List all terms.

(d) List all atomic formulas.

(e) Is the formula semantically true?

Answer.

(a) Object variables: x , y object constants: 1, 5

(b) There are two predicates in above formula: > and =
And one function +

(c) The terms in above formula are:
x, y, 1, 5, x+1

(d) The atomic formula:
x>y, x+1=5

(e) The formula is not semantically true in the domain of natural numbers. For example if x is 2 and y is 3, we can obviously see that $x > y \vee x+1=5$ is false, while the above statement says for any x and y it is true.

---

3) (10 marks) Convert the following formulae to conjunctive normal form (CNF). Clearly show **ALL** steps.

(i) $p \rightarrow q \rightarrow \neg r$

(ii) $(s \equiv r) \rightarrow (p \wedge q)$

(iii) $(\forall X)(\exists Y)lt(X,Y) \rightarrow (\exists X)(\forall Y)gt(X,Y)$

Answer.

(i)

$$p \rightarrow q \rightarrow \neg r$$
$$\Rightarrow \neg p \vee \neg q \vee \neg r$$

(ii)

$$(s \equiv r) \rightarrow (p \wedge q)$$
$$\Rightarrow \neg(s \equiv r) \vee (p \wedge q)$$
$$\Rightarrow \neg((\neg s \vee r) \wedge (\neg r \vee s)) \vee (p \wedge q)$$
$$\Rightarrow \neg(\neg s \vee r) \vee \neg(\neg r \vee s) \vee (p \wedge q)$$
$$\Rightarrow (s \wedge \neg r) \vee (r \wedge \neg s) \vee (p \wedge q)$$
$$\Rightarrow (s \wedge \neg r) \vee (r \wedge \neg s) \vee (p \wedge q)$$
$$\Rightarrow (((s \wedge \neg r) \vee r) \wedge ((s \wedge \neg r) \vee \neg s)) \vee (p \wedge q)$$
$$\Rightarrow ((s \vee r) \wedge (\neg r \vee r) \wedge (s \vee \neg s) \wedge (\neg r \vee \neg s)) \vee (p \wedge q)$$
$$\Rightarrow ((s \vee r) \wedge (\neg r \vee \neg s)) \vee (p \wedge q)$$
$$\Rightarrow ((s \vee r) \vee (p \wedge q)) \wedge ((\neg r \vee \neg s) \vee (p \wedge q))$$
$$\Rightarrow ((s \vee r \vee p) \wedge (s \vee r \vee q)) \wedge ((\neg r \vee \neg s \vee p) \wedge (\neg r \vee \neg s \vee q))$$
$$\Rightarrow (s \vee r \vee p) \wedge (s \vee r \vee q) \wedge (\neg r \vee \neg s \vee p) \wedge (\neg r \vee \neg s \vee q)$$

(iii)

$$(\forall X)(\exists Y) lt(X,Y) \rightarrow (\exists X)(\forall Y) gt(X,Y)$$
$$\Rightarrow \neg(\forall X)(\exists Y) lt(X,Y) \vee (\exists X)(\forall Y) gt(X,Y)$$
$$\Rightarrow (\exists X)\neg(\exists Y) lt(X,Y) \vee (\exists X)(\forall Y) gt(X,Y)$$
$$\Rightarrow (\exists X)(\forall Y)\neg lt(X,Y) \vee (\exists X)(\forall Y) gt(X,Y)$$
$$\Rightarrow (\exists X)(\forall Y)\neg lt(X,Y) \vee (\exists W)(\forall Z) gt(W,Z)$$
$$\Rightarrow (\exists X)(\forall Y)(\exists W)(\forall Z)\neg lt(X,Y) \vee gt(W,Z)$$
$$\Rightarrow (\forall Y)(\forall Z)\neg lt(g1,Y) \vee gt(g2,Z)$$
$$\Rightarrow \neg lt(g1,Y) \vee gt(g2,Z)$$

---

4) (10 marks) For each of the formulae in question 3,
(a) Write the formula in logic programming notation.
(b) Identify the Horn clauses in your notation in part (a). For Horn clauses, identify facts, rules, and queries.
(c) Indicate whether your notation in part (a) results in a definite program or not. Why?

Answer.

For formula (i):

(a) Logic programming notation :- p,q,r.
(b) It is a Horn clause, and it is a query (or goal).
(c) No, since it does not consist of facts and rules. Yet, it is a query that can be answered given a definite program.

For formula (ii):

(a) Logic programming notation:

$$\begin{cases} s; r; p :- . \\ s; r; q :- . \\ p :- r, s. \\ q :- r, s. \end{cases}$$

(b) The first two clauses are NOT Horn clauses, since they have more than one positive literal. The last two clauses are Horn clauses, and they are rules.

(c) No, since it does not consist of facts and rules. The first two clauses are not definite clauses.

For formula (iii):

(a) Logic programming notation:

$$gt(g2, Z) :- lt(g1, Y).$$

(b) It is a Horn clause, and it is a rule.

(c) Yes, it can be written as a definite program since it consists of a definite clause.

---

(5) (5 marks) Using what we covered in class about arithmetic in Prolog,
(a)Write a simple predicate convert(SQm, SQcm) that converts area in square meters to area in square centimetres.
(b) Write a query that converts 5 square meters to square centimetres given the above predicate. What does Prolog return as an answer to your query? Explain why.
(c) Write a query that converts 5 square centimetres to square meters given the above predicate. What does Prolog return as an answer to your query? Explain why.

Answer.

```
(a) convert(SQm, SQcm):- SQcm is SQm * 10000.
(b) :- convert(5, X).
```

Prolog will return:
X = 50000.
To answer the above query, Prolog will resolve the query with the rule given, and will obtain a new query:
:- X is 5 * 10000.
The operator 'is' evaluates the right hand side argument to obtain 50000. Now to satisfy the query, X must be equal to 50000. Therfore Prolog returns the above mentioned answer.

```
(c) :- convert(X, 5).
```

Prolog will show an error prompt.
To answer the above query, Prolog will resolve the query with the rule given, and will obtain a new query:
:- 5 is X * 10000.

Prolog does not have a value for X to evaluate the right hand side (X is not instantiated).

---

6) (5 marks) Consider the following Boolean variables and their corresponding English phrases:
p: weather is warm
q: John is swimming
r: John is outdoors
s: John is studying at home

Given the following statements:

If the weather is warm and John is not studying at home then John is outdoors.
John is swimming provided he is outdoors.
The weather is warm.

And the following query:

John is swimming if he is not studying at home.

Use the logic programming notation and a refutation tree to find the answer to above query.

$$(p \wedge \neg s) \rightarrow r \quad \Rightarrow \quad \neg p \vee s \vee r \quad \Rightarrow \quad s; r :- p. \quad (C1)$$
$$r \rightarrow q \quad \Rightarrow \quad \neg r \vee q \quad \Rightarrow \quad q :- r. \quad (C2)$$
$$p \quad \Rightarrow \quad p :- . \quad (C3)$$

Negation of the query:

$$\neg(\neg s \rightarrow q) \quad \Rightarrow \quad \neg(s \vee q) \Rightarrow \neg s \wedge \neg q \quad \Rightarrow \quad \begin{cases} :\text{-s.} & (G1) \\ :\text{-q.} & (G2) \end{cases}$$

Resolving C1 & C2 on r gives	s, q :- p.	(R1)
Resolving R1 & C3 on p gives	s,q :-.	(R2)
Resolving R2 & G1 on s gives	q:-.	(R3)
Resolving R3 & G2 on q gives	:-	

The answer is TRUE since refutation could prove inconsistency by obtaining an empty clause through resolution/refutation.

---

7) (10 marks) Given the following facts and rules:

The basic parts of a bike are front frame, back frame, and saddle.
A subpart of a bike is a part of a basic part.
A handle and a fork are parts of the front frame.

the following program is coded in Prolog (numbered from C0 to C5):

C0: basicpart(front).
C1: basicpart(back).
C2: basicpart(saddle).
C3: subpart(X) :- basicpart(Y), partof(X,Y).

C4: partof(handle, front).
C5: partof(fork, front).

Draw the search tree based on linear refutation for the following query (let's call it G0):

:- subpart(X).

Use the above labels (C0 - C5 and G0) in your search tree. Label all branches to show resolution and unifier. Under each leaf node, mention what will happen and why. If there are any outputs by Prolog, indicate them too.

Answer: