Exam-2026-01-30-exam-mod2

Tuesday, February 17, 2026 11:59 AM

1. (5 points) Consider the language of propositional logic. Use natural deduction to prove that the following holds, or find a counter-example to show that it does not hold (remember that $\neg F$ is only a shorthand for $F \to \bot$).
   - $\vdash ((A \to B) \land (B \to \neg A)) \to (A \to \bot)$
   - $\vdash (A \land B \land C) \to ((A \to B) \land (B \to C))$

$$
\begin{prooftree}
\AxiomC{$[A]_1$}
\AxiomC{$2[(A\to B)\land(B\to \neg A)]$}
\RightLabel{$(\land E)$}
\UnaryInfC{$A\to B$}
\LeftLabel{$(\to E)$}
\BinaryInfC{B}
\AxiomC{$2[(A\to B)\land(B\to \neg A)]$}
\RightLabel{$(\land E)$}
\UnaryInfC{$B\to \neg A$}
\RightLabel{$(\to E)$}
\BinaryInfC{$\neg A$}
\AxiomC{$[A]_1$}
\RightLabel{$(\to I)_1$}
\BinaryInfC{$\bot$}
\RightLabel{$(\to I)_1$}
\UnaryInfC{$A\to\bot$}
\RightLabel{$(\to I)_2$}
\UnaryInfC{$((A\to B)\land(B\to\neg A))\to(A\to\bot)$}
\end{prooftree}
$$

$$
\begin{prooftree}
\AxiomC{$[A\land B\land C]_3$}
\LeftLabel{$(\land E)$}
\UnaryInfC{$B\land C$}
\AxiomC{$[A]_2$}
\LeftLabel{$(\land I)$}
\BinaryInfC{$A\land B\land C$}
\LeftLabel{$(\land E)$}
\UnaryInfC{B}
\RightLabel{$(\to I)_1$}
\UnaryInfC{$A\to B$}
\AxiomC{$[A\land B\land C]_3$}
\LeftLabel{$(\land E)$}
\UnaryInfC{$A\land C$}
\AxiomC{$[B]_2$}
\LeftLabel{$(\land I)$}
\BinaryInfC{$A\land B\land C$}
\LeftLabel{$(\land E)$}
\UnaryInfC{C}
\RightLabel{$(\to I)_2$}
\UnaryInfC{$B\to C$}
\LeftLabel{$(\land I)$}
\BinaryInfC{$(A\to B)\land(B\to C)$}
\RightLabel{$(\to I)_3$}
\UnaryInfC{$(A\land B\land C)\to((A\to B)\land(B\to C))$}
\end{prooftree}
$$

2. (5 points) Transform and simplify the following propositional logic formula into an equivalent formula in Disjunctive Normal Form:

$$((A \leftrightarrow B) \to C) \land (D \to (A \land \neg B))$$

$$((A \leftrightarrow B) \rightarrow C) \land (D \rightarrow (A \land \neg B))$$

$$(\neg((A \rightarrow B) \land (B \rightarrow A)) \lor C) \land (\neg D \lor (A \land \neg B)) \qquad X = (A \land \neg B)$$

$$(\neg((\neg A \lor B) \land (\neg B \lor A)) \lor C) \land (\neg D \lor X)$$

$$((A \land \neg B) \lor (B \land \neg A) \lor C) \land (\neg D \lor X)$$

$$(X \lor (B \land \neg A) \lor C) \land (\neg D \lor X)$$

$$((X \lor (B \land \neg A) \lor C) \land \neg D) \lor ((X \lor (B \land \neg A) \lor C) \land X)$$

$$((X \lor (B \land \neg A) \lor C) \land \neg D) \lor X$$

$$(X \land \neg D) \lor (B \land \neg A \land \neg D) \lor (C \land \neg D) \lor X$$

$$(A \land \neg B) \lor (B \land \neg A \land \neg D) \lor (C \land \neg D)$$

3. (5 points)

4. Consider the following statements:

(a) Marco will pass the exam unless he fails to submit the assignment.
(b) Marco studies whenever he wants to pass the exam.
(c) Marco wants to pass the exam only if he studies.
(d) Marco does not submit the assignment unless he studies.

Task.

(a) Introduce appropriate propositional variables.
(b) Formalize each statement in propositional logic.
(c) Identify whether any pair of statements together expresses an " if and only if" relation.

a) P: MARCO PASS THE EXAM

A: MARCO SUBMIT THE ASSIGMENT

S: MARCO STUDY

W: MARCO WANT TO PASS

b)

1) $\neg(\neg A) \rightarrow P$ $A \rightarrow P$

2) $W \rightarrow S$

3) $W \rightarrow S$

4) $\neg S \rightarrow \neg A$ $A \rightarrow S$

c) THERE IS NOT ANY $((X \rightarrow Y) \wedge (Y \rightarrow X))$

SO THERE IS NOT ANY $\leftrightarrow$

5. (5 points) Consider the following statements:

(a) Every man trusts only persons he knows.

(b) Every honest person is trusted by everyone.

(c) For any two persons x and y, knowing y is a sufficient and necessary condition for x to trust y.

(d) There exists a woman who does not know anyone she does not trust

Task.

(a) Introduce an appropriate FOL language

(b) Formalize each statement in FOL

MAN(X) x is a man

PERSON(X) x is a person

TRUST(X,Y) X TRUST Y

KNOWS(X,Y) X KNOWS Y

HONEST(X) X is honest

HONEST(X)   X is honest

WOMAN(X)   X is a woman

a) $\forall x \forall y ((MAN(X) \land TRUST(X,Y)) \rightarrow (PERSON(Y) \land KNOWS(X,Y)))$

b) $\forall x\ (HONEST(X) \rightarrow \forall y\ TRUST(Y, X))$

c) $\forall x \forall y\ (KNOW(X,Y) \leftrightarrow TRUST(X,Y))$

d) $\exists x ((WOMAN(X) \land \forall y\ KNOW(X,Y)) \rightarrow TRUST(X,Y))$

6. **(6 points)** In a simple programming language, a program consists of a list of instructions. Some instructions "use" variables (e.g., `use(x)`). A compiler pass must verify that every variable used in the program is present in a list of declared variables. Write a recursive Prolog predicate `verify_vars(Program, DeclaredVars)` that succeeds only if every `use(Var)` instruction in the Program list corresponds to a variable found in the DeclaredVars list. **Example Queries:**

```
?- verify_vars([begin, use(a), use(b), end], [a, b, c]).
true.

?- verify_vars([use(a), use(z)], [a, b]).
false.
```

Write the Prolog code for the `verify_vars/2` predicate below.

```
member(X, [X|_])
member(X, [_|TAIL]) :- member(X, TAIL)

verify([], [])

% SIMPLE INSTRUCTION
verify([INSTRUCTION | RESTOFTHEPROGRAM], VARIABLES) :-
        INSTRUCTION \= USE(_),
        verify(RESTOFTHEPROGRAM, VARIABLES)

verify([USE(X) | RESTOFPROGRAM], VARIABLES) :-
        member(X, VARIABLES),
```

```
    member(X, VARIABLES),
    verify(BESTOFPROGRAM, VARIABLES)

verify([usc(X) | BESTOFPROGRAM], [X | VARIABLES]) :-
    \+ member(X, VARIABLES),
    verify(BESTOFPROGRAM, VARIABLES)
```

7. (6 points) A network administrator must distribute exactly **100 Gigabytes (GB)** of traffic across three servers: **Server A**, **Server B**, and **Server C**. Write a CLP program to determine valid traffic assignments based on the following constraints:

   (a) **Domain:** Each server must handle between 10 and 80 GB of traffic.

   (b) **Server A:** Must handle a traffic load that is a multiple of 10.

   (c) **Server B:** Must handle at least 10 GB *more* than Server A.

   (d) **Server C:** Must handle exactly half the traffic of Server B.

   (e) **Total:** The combined traffic of all three servers must be exactly 100 GB.

   Write the full Prolog/MiniZinc code defining the predicate `traffic_dist(A, B, C)`.

```
var [10..80] : A
var [10..80] : B
var [10..80] : C
var INT : TOTAL_TRAFFIC = A+B+C

CONSTRAINT   TOTAL_TRAFFIC = 100
CONSTRAINT   (A MOD 10) = 0
CONSTRAINT   B >= A+10
CONSTRAINT   C = (B/2)

SOLVE   SATISFY
```