Assignment9 – Propositional and First-Order Logic

Problem 9.1 (Calculi Comparison)

Prove (or disprove) the *validity* of the following *formulae* in i) *Natural Deduction* ii) *Tableau* and iii) *Resolution*:

1. $P \wedge Q \Rightarrow (P \vee Q)$

Solution:

1. ND:

(1)	1	$P \wedge Q$	Assumption
(2)	1	P	$\mathcal{ND}_0 \wedge E_l$ (on 1)
(3)	1	$P \vee Q$	$\mathcal{ND}_0 \vee I_l$ (on 2)
(4)		$P \wedge Q \Rightarrow (P \vee Q)$	$\mathcal{ND}_0 \Rightarrow I^a$ (on 1 and 3)

2. Tableau:

(1)	$(P \wedge Q \Rightarrow (P \vee Q))^F$	
(2)	$(P \wedge Q)^T$	(from 1)
(3)	$(P \vee Q)^F$	(from 1)
(4)	P^T	(from 2)
(5)	Q^T	(from 2)
(6)	P^F	(from 3)

3. Resolution: $P \wedge Q \Rightarrow (P \vee Q)$: We *negate* and build a *CNF*:

$$P \wedge Q \wedge \neg(P \vee Q)$$
$$\equiv P \wedge Q \wedge \neg P \wedge \neg Q$$

yielding clauses $\{P^T\}, \{Q^T\}, \{P^F\}, \{Q^F\}$

---

2. $(A \vee B) \wedge (A \Rightarrow C) \wedge (B \Rightarrow C) \Rightarrow C$

Solution:

1. ND:

(1)	1	$(A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C)$	*Assumption*
(2)	1	$A \lor B$	$\mathcal{ND}_0 \land E_l$ (on 1)
(3)	1	$(A \Rightarrow C) \land (B \Rightarrow C)$	$\mathcal{ND}_0 \land E_r$ (on 1)
(4)	1	$A \Rightarrow C$	$\mathcal{ND}_0 \land E_l$ (on 3)
(5)	1	$B \Rightarrow C$	$\mathcal{ND}_0 \land E_r$ (on 3)
(6)	1,6	A	*Assumption*
(7)	1,6	C	$\mathcal{ND}_0 \Rightarrow E$ (on 4 and 6)
(8)	1,8	B	*Assumption*
(9)	1,8	C	$\mathcal{ND}_0 \Rightarrow E$ (on 5 and 8)
(10)	1	C	$\mathcal{ND}_0 \lor E$ (on 2, 7 and 9)
(11)		$(A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C) \Rightarrow C$	$\mathcal{ND}_0 \Rightarrow I^a$ (on 1 and 10)

2. *Tableau*:

(1)	$((A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C) \Rightarrow C)^{\mathsf{F}}$	
(2)	$((A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C))^{\mathsf{T}}$	(from 1)
(3)	C^{F}	(from 1)
(4)	$(A \lor B)^{\mathsf{T}}$	(from 2)
(5)	$(A \Rightarrow C)^{\mathsf{T}}$	(from 2)
(6)	$(B \Rightarrow C)^{\mathsf{T}}$	(from 2)
(7)	A^{T} / $A^{\mathsf{T}} \mid C^{\mathsf{T}}$ ‖ (split on 5) — B^{T} / $B^{\mathsf{F}} \mid C^{\mathsf{T}}$ ‖ (split on 6)	(split on 4)

3. Resolution:
$(A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C) \Rightarrow C$: We *negate* and build a *CNF*:

$$(A \lor B) \land (A \Rightarrow C) \land (B \Rightarrow C) \land \neg C$$
$$\equiv (A \lor B) \land (\neg A \lor C) \land (\neg B \lor C) \land \neg C$$

yielding *clauses* $\{A^{\mathsf{T}}, B^{\mathsf{T}}\}, \{A^{\mathsf{F}}, C^{\mathsf{T}}\}, \{B^{\mathsf{F}}, C^{\mathsf{T}}\}, \{C^{\mathsf{F}}\}$.
Resolving yields:

$$\{A^{\mathsf{F}}, C^{\mathsf{T}}\} + \{C^{\mathsf{F}}\} \Longrightarrow \{A^{\mathsf{F}}\}$$
$$\{B^{\mathsf{F}}, C^{\mathsf{T}}\} + \{C^{\mathsf{F}}\} \Longrightarrow \{B^{\mathsf{F}}\}$$
$$\{A^{\mathsf{T}}, B^{\mathsf{T}}\} + \{A^{\mathsf{F}}\} \Longrightarrow \{B^{\mathsf{T}}\}$$
$$\{B^{\mathsf{T}}\} + \{B^{\mathsf{F}}\} \Longrightarrow \emptyset$$

---

3. $((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$

Solution:

1. ND:

(1)		$P \vee \neg P$	TND
(2)	2	P	*Assumption*
(3)	2,3	$(P \Rightarrow Q) \Rightarrow P$	Assumption
(4)	2	$((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$	$\mathcal{ND}_0 \Rightarrow I^a$ (on 3 and 2)
(5)	5	$\neg P$	*Assumption*
(6)	5,6	$(P \Rightarrow Q) \Rightarrow P)$	Assumption
(7)	5,6,7	P	*Assumption*
(8)	5,6,7	F	FI (on 5 and 7)
(9)	5,6,7	Q	FE (on 8)
(10)	5,6	$P \Rightarrow Q$	$\mathcal{ND}_0 \Rightarrow I^a$ (on 7 and 9)
(11)	5,6	P	$\mathcal{ND}_0 \Rightarrow E$ (on 6 and 10)
(12)	5	$((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$	$\mathcal{ND}_0 \Rightarrow I^a$ (on 6 and 11)
(13)		$((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$	$\mathcal{ND}_0 \vee E$ (on 1, 4 and 12)

2. Tableau:

(1)	$(((P \Rightarrow Q) \Rightarrow P) \Rightarrow P)^{\mathsf{F}}$		
(2)	$((P \Rightarrow Q) \Rightarrow P)^{\mathsf{T}}$		(from 1)
(3)	P^{F}		(from 1)
(4)	$(P \Rightarrow Q)^{\mathsf{F}}$	P^{T}	(split on 2)
(5)	P^{T} (from 4)		

3. Resolution:
   $((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$: We *negate* and build a *CNF*:

$$
\begin{aligned}
&((P \Rightarrow Q) \Rightarrow P) \wedge \neg P \\
\equiv&(\neg(P \Rightarrow Q) \vee P) \wedge \neg P \\
\equiv&(P \wedge \neg Q \vee P) \wedge \neg P \\
\equiv&(P \vee P) \wedge (\neg Q \vee P) \wedge \neg P
\end{aligned}
$$

   yielding clauses $\{P^{\mathsf{T}}\}, \{Q^{\mathsf{F}}, P^{\mathsf{T}}\}, \{P^{\mathsf{F}}\}$.

---

4. Can you identify any advantages or disadvantage of the *calculi*, and in which situations?

Problem 9.2 (Equivalence of CSP and SAT)

We consider

- *CSPs* $\langle V, D, C\rangle$ with *finite domains* as before
- *SAT problems* $\langle V, A\rangle$ where V is a set of propositional variables and A is a propositional formula over V.

We will show that these problem classes are equivalent by reducing their instances to each other.

1. Given a SAT instance $P = \langle V, A \rangle$, define a CSP instance $P' = \langle V', D', C' \rangle$ and two *bijections*:
   - f mapping satisfying assignments of P to solutions of P',
   - and f' the inverse of f.

   We already know that *constraint networks* are equivalent to *higher-order CSPs*. Therefore, it is sufficient to give a *higher-order CSP*.

   Solution: We define P' by $V' = V$, $D_v = \{\mathtt{T}, \mathtt{F}\}$ for every $v \in V$, and $C = \{A\}$, i.e., C contains the single *higher-order constraint* that holds if an assignment to V' (seen as an propositional assignment to V) satisfies A.
   f and f' are the identity.

   ---

2. Given a CSP instance $\langle V, D, C \rangle$, define a SAT instance (V', A') and *bijections* as above.

   Solution: We define P' as follows. V' contains variables p'_{va} for every $v \in V$ and $a \in D_v$. The intuition behind p'_{va} is that v has value a.
   A' is the conjunction of the following formulas:
   - for all $v \in V$ with $D_v = \{a_1, \dots, a_n\}$, the formula $p'_{va_1} \vee \dots \vee p'_{va_n}$ (i.e., v must have at least one value)
   - for all $v \in V$, and $a, b \in D_v$ with $a \neq b$, the formula $p'_{va} \Rightarrow \neg p'_{vb}$ (i.e., v can have at most one value)
   - for all C_{vw} and $(a, b) \notin C_{vw}$, the formula $\neg(p'_{va} \wedge p'_{wb})$ (i.e., every constraint must be satisfied)

   The *bijection* f maps a *solution* α of P to a *A'-satisfying* propositional *assignment* φ for V' as follows: for all v, a, we put $\varphi(p'_{va}) = \mathtt{T}$ if $\alpha(v) = a$ and $\varphi(p'_{va}) = \mathtt{F}$ otherwise.
   The *inverse bijection* f' maps an A'-satisfying assignment φ to a solution α of P as follows: for all v we put $\alpha(v) = a$ where a is the unique value for which $\varphi(p'_{va}) = \mathtt{T}$.

Problem 9.3 (Induction)

Use structural induction on terms and formulas to define a function C that maps every term/formula to the number of *free variable occurrences*. For example, $C(\forall x.P(x, x, y, y, z)) = 3$ because the argument has 2 *free occurrences* of y and one of z.

Hint: Use an auxiliary function $C'(V, A)$ that takes the set V of bound variables and a term/formula A. Define C' by structural induction on A. Then define $C(A) = C'(\emptyset, A)$.

Solution: C' is defined as follows for terms

- variables X: $C'(V, X) = 0$ if $X \in V$ and $C'(V, X) = 1$ if $X \notin V$
- applications of n-ary function symbol f: $C'(V, f(t_1, ..., t_n)) = \Sigma_i C'(V, t_i)$

and for formulas

- applications of n-ary predicate symbol p: $C'(V, p(t_1, ..., t_n)) = \Sigma_i C'(V, t_i)$
- nullary connectives: $C'(V, T) = C'(V, F) = 0$
- unary connectives: $C'(V, \neg A) = C'(V, A)$
- binary connectives: $C'(V, A_1 \wedge A_2) = C'(V, A_1 \vee A_2) = C'(V, A_1 \Rightarrow A_2) = C'(V, A_1) + C'(V, A_2)$
- quantifiers: $C'(V, \forall x.A) = C'(V, \exists x.A) = C'(V \cup \{x\}, A)$

This definition exhibits the typical pattern of structural induction:

- An additional argument (V) is used to track the bound variables.
- When recursing into a quantifier that argument is updated by adding the bound variable x. (In general, additional information about could be added, e.g., whether it is bound by $\forall$ or $\exists$.)
- At the leafs of the syntax tree (the base cases of the induction, here the variables), the additional argument is used.
- The main function is defined by initializing the additional argument (here with $\emptyset$).

Problem 9.4 (First-Order Semantics)

Let $= \in \Sigma_2^p$, $P \in \Sigma_1^p$ and $+ \in \Sigma_2^f$. We use the semantics of first-order logic without equality.

Prove or refute the following formulas semantically. That means you must show that $I_\varphi(A) = T$ for all models I and assignments φ (without using a proof calculus) or to give some I, φ such that $I_\varphi(A) = F$.

1. $P(X)$

Solution: Not valid. One out of many counter-examples is given by domain $\mathbb{N}$, $I(P) = \{0\}$, and $\varphi(X) = 1$.

---

2. $\forall X.\forall Y. = (+(X,Y), +(Y,X))$

Solution: Not valid. A counter-model is $\mathcal{I}_\varphi(=) = \emptyset$ with an arbitrary domain.

---

3. $\exists X.P(X) \Rightarrow (\forall Y.P(Y))$

Solution: Valid:

$$\begin{aligned}
&\mathcal{I}_\varphi(\exists X.P(X) \Rightarrow (\forall Y.P(Y)) = \top\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(P(a) \Rightarrow (\forall Y.P(Y)))) = \top\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(\neg(P(a) \wedge \neg(\forall Y.P(Y)))) = \top\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(P(a) \wedge \neg(\forall Y.P(Y))) = \bot\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(P(a)) = \bot \text{ or } \mathcal{I}_\varphi(\neg(\forall Y.P(Y))) = \bot\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(P(a)) = \bot \text{ or } \mathcal{I}_\varphi(\forall Y.P(Y)) = \top\\
\Leftrightarrow&\text{There is some } a \in \mathcal{D}_\mathcal{I} \text{ s.t. } \mathcal{I}_\varphi(P(a)) = \bot \text{ or for all } b \in \mathcal{D}_\mathcal{I} : \ \mathcal{I}_\varphi(P(b)) = \top
\end{aligned}$$

---

4. $P(Y) \Rightarrow (\exists X.P(X))$

Solution: Now the last statement holds because if the left side does not hold, then the right side must hold.
