CS 228M : Logic for Computer Science Spring 2026

Practice Problem Set 4 Solutions

---

1. **Well-formed or not?**

Solution:

Formation Rules

Terms

(T1) Every variable is a term.

(T2) If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term.

Atomic Formulas

(A1) If R is an n-ary relation symbol and $t_1, \ldots, t_n$ are terms, then $R(t_1, \ldots, t_n)$ is an atomic formula.

Formula Formation

(F1) If φ and ψ are formulas, then $(\varphi \land \psi)$ and $(\varphi \to \psi)$ are formulas.

(Q1) If φ is a formula and x is a variable, then $\forall x\varphi$ and $\exists x\varphi$ are formulas.

---

1. First Expression

$$\forall x\big(R(x,x) \to \exists y(P(x,y) \land P(f(x),y))\big)$$

Signature:

- P binary relation
- R binary relation
- f unary function
- x, y variables

Step 1: Terms

By (T1), x and y are terms.
Since f is unary and x is a term, by (T2),

$$f(x)$$

is a term.

Step 2: Atomic Formulas

Since R is binary and x is a term:

$$R(x, x)$$

is an atomic formula by (A1).
Since P is binary:

$$P(x, y)$$

is atomic because both x and y are terms.
Also,

$$P(f(x), y)$$

is atomic because $f(x)$ and y are terms.

Step 3: Build Compound Formulas

Since $P(x, y)$ and $P(f(x), y)$ are formulas, by (F1):

$$(P(x, y) \wedge P(f(x), y))$$

is a formula.
Since this is a formula and y is a variable, by (Q1):

$$\exists y(P(x, y) \wedge P(f(x), y))$$

is a formula.
Since $R(x, x)$ and the above formula are formulas, by (F1):

$$R(x, x) \to \exists y(P(x, y) \wedge P(f(x), y))$$

is a formula.
Finally, by (Q1):

$$\forall x\big(R(x, x) \to \exists y(P(x, y) \wedge P(f(x), y))\big)$$

is a formula.

Conclusion

$$\boxed{\text{The first expression is a well-formed formula (WFF).}}$$

---

2. Second Expression

$$\forall x\big(R(x, x) \to \exists f(y)(P(x, y) \wedge P(f(x)), y)\big)$$

Step 1: Quantifier Check

Rule (Q1) allows quantification only over variables.
But here we have:

$$\exists f(y)$$

Since:

- f is a function symbol (not a variable),
- $f(y)$ is a term (not a variable),

this violates the quantifier formation rule.
Therefore the formula already fails at this stage.

Conclusion

$$\boxed{\text{The second expression is NOT a well-formed formula.}}$$

2. **Additive Identity**

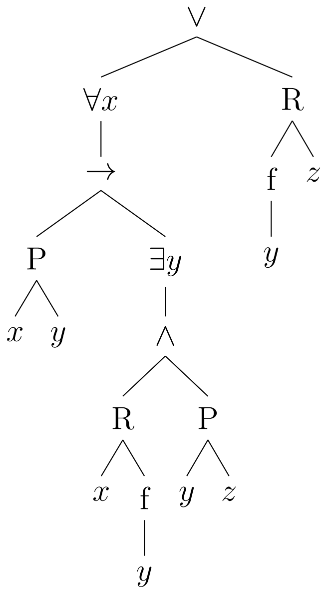
Solution:

1. $\varphi_1 \equiv \exists t \forall u\ ((u+t)=u) \land (t+u)=u)$
2. $\varphi_1 \ \rightarrow\ \varphi_2$, where $\varphi_2 \equiv \forall s \forall t\ (\forall u\ (((u+s)=u) \land ((s+u)=u) \land ((u+t)=u) \land ((t+u)=u)) \ \rightarrow\ (s=t))$
3. $\varphi_1 \land \varphi_2 \ \rightarrow\ \forall t \forall u\ (\forall v\ (((v+t)=v) \land ((t+v)=v) \rightarrow ((t \times u)=t))$

3. **FOL Parsing**

Solution:

1. **Parse Tree:**



2. **Free Variable Analysis:** All occurrences of x are **bound** by the $\forall x$ quantifier in the left branch. The first occurrence of y (in $P(x, y)$) is **free** as it is outside the scope of the $\exists y$ quantifier, and $\forall x$ does not bind y. The second and third occurrences of y (in $R(x, f(y))$ and $P(y, z)$) are **bound** by the $\exists y$ quantifier. The fourth occurrence of y (in $R(f(y), z)$) is **free**, as it is outside the scope of both quantifiers. All occurrences of z are **free**. Because there are free variables (specifically y and z) in φ, **it is not a sentence**.

3. **Substitution:**
   We must determine if the given terms can be substituted for the *free* occurrences of y. A given term can be substituted for the free occurrences of a variable if no variable of the term gets bounded by its quantifier.

   The free occurrences of y are located inside $P(x, y)$ (which is under the scope of $\forall x$) and $R(f(y), z)$ (which is under no quantifier scope).

   (a) $t_2 = f(z)$**: Substitutable.**
   Substituting $f(z)$ for the free occurrences of y results in $P(x, f(z))$ and $R(f(f(z)), z)$. The term $f(z)$ contains the variable z. Because z does not get bound by the $\forall x$ quantifier that scopes over $P(x, y)$, no variable capture occurs.

   (b) $t_3 = f(x)$**: Not substitutable.**
   Substituting $f(x)$ for the free occurrences of y results in replacing $P(x, y)$ with $P(x, f(x))$. Because this specific occurrence of y is within the scope of the $\forall x$ quantifier, the x inside the incoming term $f(x)$ gets bound by $\forall x$.

4. **Groups and logic**

Solution:

(a) **Associativity of the operation** $+$

This property states that the grouping of elements does not affect the result of the operation. For any elements x, y, and z in G, performing $(x + y) + z$ should give the same result as $x + (y + z)$.

$$\forall x \forall y \forall z \; ((x + y) + z = x + (y + z))$$

(b) **Right identity and right inverse**

A right identity means that adding 0 to the right of any element leaves the element unchanged. A right inverse means that for every element x in G, there exists some element y such that their operation results in the identity element 0.

Right identity:

$$\forall x \; (x + 0 = x)$$

Right inverse:

$$\forall x \; \exists y \; (x + y = 0)$$

(c) **Right cancellation property**

This property states that if adding the same element z to two elements x and y results in the same value, then x and y must be equal.

$$\forall x \forall y \forall z \; ((x + z = y + z) \rightarrow (x = y))$$

5. **Blood Relations**

Solution:

1. (a)

$$Mother(m, c) \equiv Parent(m, c) \wedge Female(m)$$

(b)

$$Sibling(x, y) \equiv x \neq y \wedge \exists p\big(Parent(p, x) \wedge Parent(p, y)\big)$$

(c)

$$Grandparent(g, c) \equiv \exists p\big(Parent(g, p) \wedge Parent(p, c)\big)$$

2. (a)

$$\forall x \; \exists m \Big(Parent(m, x) \wedge Female(m) \; \wedge \; \forall t \big((Parent(t, x) \wedge Female(t)) \rightarrow t = m \big) \Big)$$

The statement says every person has exactly one mother.

This requires two conditions:

- Existence: every person x must have at least one mother.
- Uniqueness: no person can have two different mothers.

The part

$$\exists m(Parent(m, x) \wedge Female(m))$$

ensures that a mother exists.

The part

$$\forall t((Parent(t,x) \wedge Female(t)) \rightarrow t = m)$$

ensures uniqueness: if another person t is also a female parent of x, then t must be equal to m.

Thus m is the unique mother of x.

3. Matriarch condition:

$$Female(m) \wedge \neg\exists p\, Parent(p,m)$$

Final sentence:

$$\forall x\exists m(Female(m) \wedge \neg\exists p Parent(p,m) \wedge (x = m \vee Parent(m,x) \vee \exists z(Parent(m,z) \wedge Parent(z,x))))$$

This basically ensures that for every person there is atleast one matriarch women who is either his parent or grandparent.

6. **Spot the difference**

Solution: A key structural difference is that in G_1 there exists a vertex with two distinct outgoing edges (vertex a points to both b and c), whereas in G_2 every vertex has exactly one outgoing edge.

This property can be expressed by the following sentence:

$$\exists x\exists y\exists z\big(R(x,y) \wedge R(x,z) \wedge y \neq z\big)$$

Explanation:

The formula states that there exists a vertex x that has edges to two distinct vertices y and z. In G_1, vertex a satisfies this property since $a \to b$ and $a \to c$ with $b \neq c$. Thus the sentence is true in G_1.

In G_2, every vertex has exactly one outgoing edge, so no vertex can point to two distinct vertices. Therefore the sentence is false in G_2.

Hence this formula distinguishes the two graphs.

7. **Prefix Order on Binary Strings**

Solution:

(a) A total order requires that every pair of elements be comparable, i.e.

$$\forall x\forall y\ (\mathsf{Pref}(x,y) \vee \mathsf{Pref}(y,x)).$$

Therefore, to ensure that the relation is *not* a total order, we must require that there exist two incomparable elements. The required condition is:

$$\exists x\exists y\ \big(\neg\mathsf{Pref}(x,y) \wedge \neg\mathsf{Pref}(y,x)\big).$$

This says that there are two elements x and y such that neither is a prefix of the other, so the order is not total.

(b) We first note that "x is a proper prefix of y" means:

$$\mathsf{Pref}(x, y) \land x \neq y.$$

Now y is an immediate extension of x if:

- x is a proper prefix of y, and
- there is no string z strictly between x and y in the prefix order.

So a suitable formula is:

$$\mathsf{Imm}(x, y) := \mathsf{Pref}(x, y) \land x \neq y \land \neg\exists z \ \big(\mathsf{Pref}(x, z) \land x \neq z \land \mathsf{Pref}(z, y) \land z \neq y\big).$$

(c) We want to say that for every string x, there are exactly two distinct strings y and z such that both are immediate extensions of x, and every immediate extension of x is one of them.

Hence the required sentence is:

$$\forall x \ \exists y \exists z \ \Big(y \neq z \land \mathsf{Imm}(x, y) \land \mathsf{Imm}(x, z) \land \forall w \ (\mathsf{Imm}(x, w) \to (w = y \lor w = z))\Big).$$

This ensures:

- at least two immediate extensions, namely y and z,
- y and z are distinct,
- and there are no immediate extensions other than these two.

In the intended structure $\mathcal{B}$, these are exactly the two strings obtained by appending 0 and 1 to x.

8. **Bit-Vectors**

Solution:

(a) Bitwise Equality

$$\forall i \ (S_x(i) \leftrightarrow S_y(i))$$

(b) Common Zeros

$$\exists i \ (\neg S_x(i) \land \neg S_y(i))$$

(c) Cardinality (The "Singleton" Property)

$$\exists i \ (S_x(i) \land \forall j \ (S_x(j) \implies j = i))$$

(d) Mind the gap

$$\exists i \exists j \exists k \ \big(< (i, j) \land < (j, k) \land S_x(i) \land S_x(k) \land \neg S_x(j) \land \neg\exists l(< (i, l) \land < (l, k) \land \neg(k = l))\big) \ \land$$
$$\forall u \forall v \forall w \ \neg\big(< (u, v) \land < (v, w) \land S_y(u) \land S_y(w) \land \neg S_y(v) \land \neg\exists p(< (u, p) \land (p, w) \land \neg(p = v))\big)$$