Exercises 3

Exercise 1 (Skolemization & Herbrand Universes)**.** The following formulas (on the signature $\{P, E\}$ with two predicate symbol and $ar(P) = ar(E) = 2$) form the theory of Unbounded Dense Total Orders. The set $\mathbb{R}$ of real numbers is an example of such an order where $P(x, y)$ denotes $x < y$ and $E(x, y)$ denotes $x = y$. For each of the axioms, apply *negation normal form*, *Skolemization* and *prenex normal form* (in this order). **Solution:**

Axiom:	$\forall x.\, E(x, x)$
NNF:	$\forall x.\, E(x, x)$
Skolemization:	$\forall x.\, E(x, x)$
PNF:	$E(x, x)$

Axiom:	$\forall x, y, z.\, (P(x, y) \land P(y, z)) \to P(x, z)$
NNF:	$\forall x, y, z.\, \neg P(x, y) \lor \neg P(y, z) \lor P(x, z)$
Skolemization:	$\forall x, y, z.\, \neg P(x, y) \lor \neg P(y, z) \lor P(x, z)$
PNF:	$\neg P(x, y) \lor \neg P(y, z) \lor P(x, z)$

Axiom:	$\forall x, y.\, P(x, y) \to \exists z.\, P(x, z) \land f(z, y)$
NNF:	$\forall x, y.\, \neg P(x, y) \lor \exists z.\, P(x, z) \land f(z, y)$
Skolemization:	$\forall x, y.\, \neg P(x, y) \lor P(x, f_z(x, y)) \land f(f_z(x, y), y)$
PNF:	$\neg P(x, y) \lor P(x, f_z(x, y)) \land f(f_z(x, y), y)$

Axiom:	$\forall x, y.\, E(x, y) \leftrightarrow \neg(P(x, y) \lor P(y, z))$
NNF:	$\forall x, y.\, (\neg E(x, y) \lor (\neg P(x, y) \land \neg P(y, z))) \land (P(x, y) \lor P(y, z) \lor E(x, y))$
Skolemization:	$\forall x, y.\, (\neg E(x, y) \lor (\neg P(x, y) \land \neg P(y, z))) \land (P(x, y) \lor P(y, z) \lor E(x, y))$
PNF:	$(\neg E(x, y) \lor (\neg P(x, y) \land \neg P(y, z))) \land (P(x, y) \lor P(y, z) \lor E(x, y))$

Axiom:	$\forall x, y, z.\, (E(x, y) \land E(y, z)) \to E(x, z)$
NNF:	$\forall x, y, z.\, \neg E(x, y) \lor \neg E(y, z) \lor E(x, z)$
Skolemization:	$\forall x, y, z.\, \neg E(x, y) \lor \neg E(y, z) \lor E(x, z)$
PNF:	$\neg E(x, y) \lor \neg E(y, z) \lor E(x, z)$

Axiom:	$\forall x, y, z.\, (E(x, y) \land P(x, z)) \to P(y, z)$
NNF:	$\forall x, y, z.\, \neg E(x, y) \lor \neg P(x, z) \lor P(y, z)$
Skolemization:	$\forall x, y, z.\, \neg E(x, y) \lor \neg P(x, z) \lor P(y, z)$
PNF:	$\neg E(x, y) \lor \neg P(x, z) \lor P(y, z)$

Axiom:	$\forall x, y, z.\, (E(x, y) \wedge P(z, x)) \rightarrow P(z, y)$
NNF:	$\forall x, y, z.\, \neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$
Skolemization:	$\forall x, y, z.\, \neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$
PNF:	$\neg E(x, y) \vee \neg P(z, x) \vee P(z, y)$

$\Diamond$

What does Herbrand's Theorem say about this set of axiom? Can you find an example? **Solution:** *Herbrand's theorem tells us that we can find a countable dense order.* $(\mathbb{Q}, <, =)$ *is such an example.* $\Diamond$

Exercise 2 (Effectively Propositional Logic)**.** Consider the class of formulas of first order logic built on a signature containing only constant symbols (arity 0 functions) and predicate symbols, and of the following form:

$$\forall x_1 \ldots \forall x_n.\ F(x_1, \ldots, x_n)$$

where F is quantifier-free.

1. Show that this set of formula is closed under conjunction, disjunction and negation for satisfiability, by which is meant that for arbitrary formulas F_1 and F_2, you can efficiently compute formulas in the above form that are equisatisfiable to $F_1 \wedge F_2$, $F_1 \vee F_2$ and $\neg F_1$.

   Solution: *The most complicated part is finding a procedure to find an formula equisat to* $\neg F_1$*.*

   $$\neg\forall x_1 \ldots \forall x_n.\ F(x_1, \ldots, x_n) \equiv \exists x_1 \ldots \exists x_n.\ \neg F(x_1, \ldots, x_n)$$

   By applying Skolemization, we can remove the first quantifier and obtain the following equisatisfiable formula:

   $$\exists x_2 \ldots \exists x_n.\ \neg F(f_{x_1}, \ldots, x_n)$$

   Where f_{x_1} *is a new constant symbol introduced by the skolemization. By repeating the process other* $n - 1$ *times we can prove by induction that* $\neg F$ *is equisat to:*

   $$\neg F(f_{x_1}, \ldots, f_{x_n})$$

   which is quantifier free and whose signature contains only constant symbols and predicates.

   The disjunction of $\forall x_1 \ldots \forall x_n.\ F_1(x_1, \ldots, x_n)$ *and* $\forall y_1 \ldots \forall y_m.\ F_2(y_1, \ldots, y_m)$ *is equisatisfiable to* $\forall x_1 \ldots \forall x_n . \forall y_1 \ldots \forall y_m.\ F_1(x_1, \ldots, x_n) \vee F_2(y_1, \ldots, y_m)$*. The same goes for conjunction.* $\Diamond$

2. Show there exists an algorithm to decide the satisfiability of such formulas.

   Solution: *First, note that since the theory is composed of one single formula (which has a finite size), one can assume that the set of relations $\mathcal{R}$ and constants $\mathcal{C}$ of the language is finite (i.e. restricted to the ones appearing in the formula at hand). Without loss of generality, suppose the language contains at least one constant symbol (possibly not appearing in the formula).*

   Suppose the formula is satisfiable, i.e. it has a model. Then, Herbrand's theorem applies and $(\mathrm{GT}_{\mathcal{L}}, \alpha_h)$ is also a model. However, since all functions have arity 0 (i.e. are constants), we have $\mathrm{GT}^i = \mathrm{GT}^j$ for any $i, j \in \mathbb{N}^*$. This means that

$$\mathrm{GT}_{\mathcal{L}} = \bigcup_{i \geq 0} \mathrm{GT}^i = \mathrm{GT}^1 = \mathcal{C}$$

   Since the set of constants is finite, so is the domain of the ground model. In particular, the formula has a model if and only if it has a model of size $|\mathcal{C}|$.

   Thus, it is sufficient to try exhaustively all possible interpretations of the relation symbols on this finite domain. There are exponentially, but finitely, many different such interpretations.

   To summarize the procedure:

   (a) *Generate the finite ground domain of the language;*

   (b) *For each possible interpretation of the relation symbols, evaluate the formula. The formula is satisfiable if and only if is evaluates to true on at least one interpretation.*

   As soon as the formula holds for all assignments in one of the interpretations, the formula is satisfiable; else, if it does in none, it is unsatisfiable.

   $\Diamond$

Exercise 3 (ITE)**.** Consider the ternary propositional operation $\mathsf{ite}(\mathsf{x},\mathsf{y},\mathsf{z})$ (if x then y else z) defined by the following truth table:

x	y	z	$\mathsf{ite}(\mathsf{x},\mathsf{y},\mathsf{z})$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

We consider expressions build only from variables and ite, without constants 0 and 1. We call them pure ite expressions.

- (1pt) Express $x \wedge y$ and $x \vee y$ as a pure ite expression. **Solution:** $\mathsf{ite}(\mathsf{x},\mathsf{y},\mathsf{x})$ *and* $\mathsf{ite}(\mathsf{x},\mathsf{x},\mathsf{y})$ $\Diamond$
- (1pt) Let e denote an ite expression whose set of free variables is $V = \{x_1, \ldots, x_n\}$. Let v be an assignment assigning all variables in V to 0 (false). What is the truth value of e under v? **Solution:** *The truth value of a pure* ite *expression is given by an evaluation function that traverses the expression from the root following a single path until it reaches the leaf of the expression, which can only be a variable. Because all variables are interpreted as 0, the resulting expression is 0. We observe that, analogously, if all variables are interpreted as 1, then the value of the expression is 1.* $\Diamond$
- (1pt) Give an example of a function $f : \{0,1\}^n \to \{0,1\}$ that is not expressible as a pure ite expression. **Solution:** *From the previous observation* $f(0,\ldots,0) = 0$ *and* $f(1,\ldots,1) = 1$ *for every function expressible as a pure* ite *expression. Thus, any function for which either* $f(0,\ldots,0) = 1$ *or for which* $f(1,\ldots,1) = 0$ *cannot be expressed using pure* ite *expression. Examples include constant functions: function* $f_0(x_1,\ldots,x_n) = 0$ *for all* x_i*, function* $f_1(x_1,\ldots,x_n) = 1$ *for all* x_i*, negation* $f(x) = \neg x$ *because, for example,* $f(0) = 1$*, and exclusive or* $f(x_1,x_2) = x_1 \oplus x_2$ *because* $f(1,1) = 0$*.* $\Diamond$
- (4pt) List (in the separate sheet) the answer numbers next to all true completions of the statement:

  Solution: *The two conditions* $f(0,\ldots,0) = 0$ *and* $f(1,\ldots,1) = 1$ *are necessary for* f *to be expressible as pure* ite *expression, as explained in the solution to C). We now show that it is also sufficient. The idea*

is that we build a binary tree making case analysis on the truth values $v_1, \ldots, v_n$ of all variables $x_1, \ldots, x_n$. Each branch of a tree is an ite *node and each leaf is given by a path that follows the values $v_1, \ldots, v_n$. In all branches except the ones corresponding to the values $(0, \ldots, 0)$ and $(1, \ldots, 1)$, there must be a variable v_i that is interpreted as 1 and a variable v_j that is interpreted as 0. If $f(v_1, \ldots, v_n) = 1$ then we use as the corresponding leaf for the path $v_1, \ldots, v_n$ the smallest x_i for which $v_i = 1$. Otherwise, we put the smallest x_j for which $v_j = 0$. Such pure* ite *expression encodes precisely the truth table of f. Remark that for a function of n variables there are 2^n assignments and 2^{2^n} boolean functions. With the restriction on two values of the function, the number of expressible functions is 2^{2^n-2}, or one quarter $(\frac{1}{4})$ of all boolean functions.* $\Diamond$

The functions $f : \{0,1\}^n \to \{0,1\}$ that can be expressed as pure ite expressions using variables $x_1, \ldots, x_n$ are precisely ...

1. the functions that are not constant, i.e., not equal to either the function $f_1(x_1, \ldots, x_n) = 1$ nor to the function $f_0(x_1, \ldots, x_n) = 0$
2. all functions if $n \geq 3$; functions expressible using $\wedge, \vee$ in cases $n = 1, 2, 3$
3. the functions that could be expressed using an expression with only $\wedge$ and $\vee$
4. the functions f such that $f(0, \ldots, 0) = 0$
5. the functions f such that $f(0, \ldots, 0) = 0$ and $f(1, \ldots, 1) = 1$
6. the functions expressible using $\neg(x \wedge y)$ as a binary operation
7. the functions such that $f(x, \ldots, x) = x$ for every x
8. the functions such that $f(x, \ldots, x, f(x, \ldots, x)) = x$ for every x

For those answers that you chose, explain briefly why they are correct.
Solution: *Statement 1 is **not** correct because it does not rule out, for example negation.*

*Statement 2 is **not** correct because it does not rule out, for example, a constant function for $n = 4$.*

*Statement 3 is **not** correct because it rules out too many functions, including the function $f(x_1, x_2, x_3) = \mathsf{ite}(\mathsf{x}_1, \mathsf{x}_2, \mathsf{x}_3)$. This function is not expressible using $\wedge, \vee$ because it is not monotonic. Indeed, functions expressible using $\wedge, \vee$ are monotonic, which means that*

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \leq f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

for all $0 \leq i \leq n$. However, the property

$$\mathsf{ite}(0, \mathsf{x}_2, \mathsf{x}_3) \leq \mathsf{ite}(1, \mathsf{x}_2, \mathsf{x}_3)$$

does not hold as it reduces to $x_3 \leq x_2$ and thus fails to hold for $x_3 = 1$, $x_2 = 0$ i.e. for

$$\mathsf{ite}(0, 0, 1) \leq \mathsf{ite}(1, 0, 1)$$

Statement 4 is not correct because it omits the requirement $f(1, \ldots, 1) = 1$ and thus would admit function $f(x) = 1$ which is not expressible.

***Statement 5** is the correct characterizations and thus one of the answers that should be indicated.*

Statement 6 is not correct because it admits all functions, as $\neg(x \wedge y)$ can express all boolean functions.

***Statement 7** is also correct. It is a universal statement over all x, but x is either 0 or 1, so it is equivalent to the conjunction $f(0, \ldots, 0) = 0$ and $f(1, \ldots, 1) = 1$.*

Statement 8 is not a correct characterization. uses a universally quantified statement. Note that if the correct characterization holds, then this property holds; the question is whether the property is strong enough. Consider $n = 2$. The case $x = 0$ becomes

$$f(0, f(0, 0)) = 0$$

The case $x = 1$ becomes

$$f(1, f(1, 1)) = 1$$

If we let f be the exclusive or $\oplus$ (addition modulo 2), then these properties are satisfied. However, exclusive or is not expressible as a pure ite *expression because $f(1, 1) = 1 \oplus 1 = 0$. $\Diamond$*

Exercise 4 (Resolution with Congruence)**.** Consider a set D with a binary relation $\equiv$ on D and a binary function $\sqcup : D^2 \to D$ which satisfy the following properties:

$$\begin{array}{ll}
\mathsf{assoc}: & x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z \\
\mathsf{transE}: & (x \equiv y) \wedge (y \equiv z) \to (x \equiv z) \\
\mathsf{sym}: & (x \equiv y) \to (y \equiv x) \\
\mathsf{congL}: & (x_1 \equiv x_2) \to (x_1 \sqcup y \equiv x_2 \sqcup y) \\
\mathsf{congR}: & (y_1 \equiv y_2) \to (x \sqcup y_1 \equiv x \sqcup y_2)
\end{array}$$

Define another binary relation $\sqsubseteq$ on D by:

$$x \sqsubseteq y \quad \leftrightarrow \quad x \sqcup y \equiv y \qquad (\mathsf{DefLE})$$

We claim that it follows that $\sqsubseteq$ is a transitive relation, that is:

$$x \sqsubseteq y \wedge y \sqsubseteq z \;\rightarrow\; x \sqsubseteq z \qquad \text{(TransLE)}$$

In other words, we wish to prove that the following holds in pure first-order logic, where all formulas are considered universally quantified:

$$\{\mathsf{assoc}, \mathsf{transE}, \mathsf{sym}, \mathsf{congL}, \mathsf{congR}, \mathsf{DefLE}\} \models \mathsf{TransLE} \qquad (*)$$

Note that, when representing the problem in first order logic, $t_1 \equiv t_2$ is just a notation for $E(t_1, t_2)$ where E is some binary predicate symbol, $t_1 \sqsubseteq t_2$ is a notation for $L(t_1, t_2)$ where L is another binary predicate symbol, and $t_1 \sqcup t_2$ is a notation for $f(t_1, t_2)$ where f is some binary function symbol. You can choose to either use notation $\equiv, \sqsubseteq, \sqcup$ or E, L, f, but use one or the other consistently.

Solution: *We first outline why this proof should exist by stating a simple result about orders and associative operations. For this preliminary statement we use the usual mathematical equality; we later expand this into a proof in first-order logic by applying appropriate properties of equality (sym, congL, congR). Let $\sqcup$ be associative and define $\sqsubseteq$ according to DefLE. To prove transitivity (TransLE), assume $x \sqsubseteq y$ and $y \sqsubseteq z$. This means $x \sqcup y = y$ and $y \sqcup z = z$. Thus, using associativity and the definition of $\sqsubseteq$,*

$$x \sqcup z = x \sqcup (y \sqcup z) = (x \sqcup y) \sqcup z = y \sqcup z = z$$

which by definition means that $x \sqsubseteq z$. The resolution proof will reflect the proof above, but needs to explicitly invoke, for example, congL and sym to conclude the first equality above. The Stainless code below hints at these steps. $\Diamond$

Our goal is to use resolution for first-order logic to derive a formal proof of $(*)$ by deriving a contradiction.

A) (2pt) To begin the proof, write down a numbered sequence of *clauses* that you will need in your proof, corresponding to $\mathsf{assoc}, \mathsf{transE}, \ldots$ (whatever the initial set of clauses should be to prove $(*)$ by contradiction).

1. $\{x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z\}$ (from assoc)
2. $\{\neg(x \equiv y), \neg(y \equiv z), (x \equiv z)\}$ (from transE)
3. ...

You may need more than one clause to encode some of the formulas.

Solution:

1. $\{\{x \sqsubseteq y\}, \{y \sqsubseteq z\}, \{\neg(x \sqsubseteq z)\}\}$ *(from TransLE)*

2. $\{\{\neg(x \sqsubseteq y), x \sqcup y \equiv y\}, \{x \sqsubseteq y, \neg(x \sqcup y \equiv y)\}\}$ *(from DefLE)*

3. $\{x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z\}$ *(from assoc)*

4. $\{\neg(x \equiv y), \neg(y \equiv z), x \equiv z\}$ *(from transE)*

5. $\{\neg(x \equiv y), y \equiv x\}$ *(from sym)*

6. $\{\neg(x_1 \equiv x_2), x_1 \sqcup y \equiv x_2 \sqcup y\}$ *(from congL)*

7. $\{\neg(y_1 \equiv y_2), x \sqcup y_1 \equiv x \sqcup y_2\}$ *(from congR)*

$\Diamond$

B) (4pt) Continue to write proof steps where each step follows from previous ones. For each step indicate from which previous steps it follows and by which substitution of variables. The last step should be the empty clause $\emptyset$. To save you time in finding the resolution proof, we provide the following fragment of a Stainless file which we used to check this fact; even if this is not a resolution proof, the invocations of functions in the proof provides hints about the substitutions that you may want to use in your proof.

```
1  def sym(x: D, y: D) = {...}.ensuring( !(x ≡ y) || y ≡ x )
2  ... // define the other axioms
3
4  def transitive(x: D, y: D, z: D) = {
5    require(x ⊑ y)
6    require(y ⊑ z)
7    sym(y ⊔ z, z)
8    congR(x, z, y ⊔ z)
9    assoc(x, y, z)
10   trans(x ⊔ z, x ⊔ (y ⊔ z), (x ⊔ y) ⊔ z)
11   congL(x ⊔ y, y, z)
12   transE(x ⊔ z, (x ⊔ y) ⊔ z, y ⊔ z)
13   transE(x ⊔ z, y ⊔ z, z)
14 }.ensuring(x ⊑ z)
```

Solution:

8. $\{x \sqsubseteq y\}$ *(by 1)*

9. $\{x \sqcup y \equiv y\}$ *(by 8 and 2 with $x := x;\ y := y$)*

10. $\{y \sqsubseteq z\}$ *(by 1)*

11. $\{y \sqcup z \equiv z\}$ *(by 10 and 2 with $x := x;\ y := y$)*

12. $\{\neg(y \sqcup z \equiv z), z \equiv y \sqcup z\}$ *(by 5 with $x := y \sqcup z;\ y := z$)*

13. $\{z \equiv y \sqcup z\}$ *(by 11 and 12)*

14. $\{\neg(z \equiv y \sqcup z), x \sqcup z \equiv x \sqcup (y \sqcup z)\}$ *(by 7 with* $x := x;\ y_1 := z;\ y_2 := y \sqcup z$*)*

15. $\{x \sqcup z \equiv x \sqcup (y \sqcup z)\}$ *(by 13 and 14)*

16. $\{x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z\}$ *(by 3 with* $x := x;\ y := y;\ z := z$*)*

17. $\{\neg(x \sqcup z \equiv x \sqcup (y \sqcup z)), \neg(x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z), x \sqcup z \equiv (x \sqcup y) \sqcup z\}$ *(by 4 with* $x := x \sqcup z;\ y := x \sqcup (y \sqcup z);\ z := (x \sqcup y) \sqcup z$*)*

18. $\{\neg(x \sqcup (y \sqcup z) \equiv (x \sqcup y) \sqcup z), x \sqcup z \equiv (x \sqcup y) \sqcup z\}$ *(by 15 and 17)*

19. $\{x \sqcup z \equiv (x \sqcup y) \sqcup z\}$ *(by 16 and 18)*

20. $\{\neg(x \sqcup y \equiv y), (x \sqcup y) \sqcup z \equiv y \sqcup z\}$ *(by 6 with* $x_1 := x \sqcup y;\ x_2 := y;\ y := z$*)*

21. $\{(x \sqcup y) \sqcup z \equiv y \sqcup z\}$ *(by 9 and 20)*

22. $\{\neg(x \sqcup z \equiv (x \sqcup y) \sqcup z), \neg((x \sqcup y) \sqcup z \equiv y \sqcup z), x \sqcup z \equiv y \sqcup z\}$ *(by 4 with* $x := x \sqcup z;\ y := (x \sqcup y) \sqcup z;\ z := y \sqcup z$*)*

23. $\{\neg((x \sqcup y) \sqcup z \equiv y \sqcup z), x \sqcup z \equiv y \sqcup z\}$ *(by 15 and 22)*

24. $\{x \sqcup z \equiv y \sqcup z\}$ *(by 21 and 23)*

25. $\{\neg(x \sqcup z \equiv y \sqcup z), \neg(y \sqcup z \equiv z), x \sqcup z \equiv z\}$ *(by 4 with* $x := x \sqcup z;\ y := y \sqcup z;\ z := z$*)*

26. $\{\neg(y \sqcup z \equiv z), x \sqcup z \equiv z\}$ *(by 25 and 24)*

27. $\{x \sqcup z \equiv z\}$ *(by 26 and 11)*

28. $\{\neg x \sqsubseteq z\}$ *(by 1)*

29. $\{\neg(x \sqcup z \equiv z)\}$ *(by 28 and 2 with* $x := x;\ y := z$*)*

30. $\{\}$ *(by 27 and 29)*

Note that this solution express all the instantiations as individual step for maximum details, but it is perfectly acceptable to do instantiation in resolution steps, which maked the proof much shorter. $\Diamond$

Exercise 5 (Logic of Partial Functions)**.** We wish to use first-order logic for our verification task, which requires proving validity of formulas. We use a first-order language (signature) with a relational symbol E, which we would like to represent equality and satisfy the laws of reflexivity, symmetry, and transitivity, as well as congruence laws: (analogous to congL and congR in Exercise 4) equal elements are related in the same way by all other relation symbols. We also need a way to represent a *partial* function $\bar{p}(x, y)$ of two arguments: the function can have at most one result, but it can be undefined for certain pairs of elements. Applying $\bar{p}$ when argument is undefined gives

undefined result. We will analyze two possibilities for encoding such partial functions in first-order logic, with questions arising in each of them.

Encoding 1. We use a constant b to represent undefined element and a binary function symbol $p(x, y)$ to represent the function $\bar{p}$ (note that we use p to represent the function symbol, whereas $\bar{p}$ represents the function that interprets it). The language of formulas in this part has only symbols $\{E, p, b\}$.

A) (1pt) Write down, as universally quantified first-order logic formulas, the congruence properties of p with respect to E, namely: if in the expression $p(x, y)$ we change x to an E-related element or change y to an E-related element, the new result is E-related to $p(x, y)$.

Solution: *A possible answer is*

$$\forall x.\forall y.\forall z.\ (E(x, z) \to E(p(x, y), p(z, y)))$$
$$\forall x.\forall y.\forall z.\ (E(y, z) \to E(p(x, y), p(x, z)))$$

$\Diamond$

B) (1pt) Write down, as a universally quantified first-order logic formula, the property that applying p when one of the arguments is undefined results in an undefined value.

Solution: *A possible answer is*

$$\forall x.\forall y.\ E(p(b, y), b) \wedge E(p(x, b), b)$$

$\Diamond$

C) (1pt) Describe Herbrand universe (the set of ground terms) in this language. Is it finite or infinite?

Solution:

The set of ground terms of this language is of the form

$$GT_{\mathcal{L}} = \{b, p(b, b), p(p(b, b), b) \dots\}$$

Formally $GT^0_{\mathcal{L}} = \emptyset \quad GT^{i+1}_{\mathcal{L}} = \{b\} \cup \{p(t_1, t_2) | t_1, t_2 \in GT^i_{\mathcal{L}}\} \quad GT_{\mathcal{L}} = \bigcup_{i=0}^{\infty} GT^i_{\mathcal{L}}$

It is countably infinite. $\Diamond$

Encoding 2. We use a ternary relation symbol $P(x, y, z)$ to represent the fact $\bar{p}(x, y) = z$ when all values are defined. To represent that the function is not defined for a given x and y, relation interpreting P would simply not contain any (interpretation of) z such that $P(x, y, z)$ is true. Let a be a

constant denoting an arbitrary element of the universe. The language of formulas in this part has only symbols $\{E, P, a\}$.

D) (1pt) Write down, as universally quantified first-order logic formulas, the congruence properties of P with respect to E, namely: if elements x, y, z are related by P, then elements E-related to them are also related by P, as expected by a relation with properties of equality.

Solution: *A possible answer is*

$$\forall x.\forall y.\forall z.\forall w.\ E(x,w) \to (P(x,y,z) \leftrightarrow P(w,y,z))$$
$$\forall x.\forall y.\forall z.\forall w.\ E(y,w) \to (P(x,y,z) \leftrightarrow P(x,w,z))$$
$$\forall x.\forall y.\forall z.\forall w.\ E(z,w) \to (P(x,y,z) \leftrightarrow P(x,y,w))$$

$\Diamond$

E) (1pt) Write down as a universally quantified formula, the property that P should represent a functional relation modulo E: for given pair of elements x, y, the result of $\bar{p}$, if it exists, is unique up to E.

Solution: *A possible answer is*

$$\forall x.\forall y.\forall z.\forall w.\ (P(x,y,z) \wedge P(x,y,w)) \to E(z,w)$$

$\Diamond$

F) (1pt) Consider the result of applying Skolemization of the properties in D) and E). How many new Skolem functions are introduced? Describe the Herbrand universe (the set of ground terms) in this language. Is it finite or infinite?

Solution:

Since there are no existential quantifiers, applying Skolemization does not introduce new function symbols in the language. Therefore, the set of ground terms in this new language is $\{a\}$ $\Diamond$

G) (3pt) Is there a terminating algorithm for checking, given two formulas F_1, F_2 in prenex form with only universal quantifiers using symbols from $\{E, P, a\}$, whether $\mathsf{Ax} \cup \{F_1\} \models F_2$ holds, where Ax are the Skolemized versions of properties introduced in D) and E). If yes, sketch the algorithm. If no, argue why the problem is undecidable.

Solution: *We claim it is sufficient to check* $\mathsf{Ax} \cup \{F_1\} \models F_2$ *on models with* $n+1$ *elements, where* n *is the number of quantifiers in* F_2.

$$\mathsf{Ax} \cup \{F_1\} \models F_2$$

is equivalent to

$$\models (\mathsf{Ax} \wedge F_1) \to F_2$$

i.e. validity of $(\mathsf{Ax} \wedge F_1) \rightarrow F_2$. *This is equivalent to the satisfiability of* $\neg((\mathsf{Ax} \wedge F_1) \rightarrow F_2)$ *which translates to*

$$\mathsf{Ax} \wedge F_1 \wedge \neg F_2$$

Writing F_2 *as* $\forall x_1, \ldots, x_n\ F_2'$, *this is again equivalent to*

$$\mathsf{Ax} \wedge F_1 \wedge \exists x_1, \ldots, x_n\ \neg F_2'$$

We can now skolemize the expression to obtain an equisatisfiable formula:

$$\mathsf{Ax} \wedge F_1 \wedge \neg F_2'[x_1 := c_1, \ldots, x_n := c_n]$$

This formula has $n + 1$ *constant symbols* (c_1 *to* c_n *and* a) *and no function symbol, so its Herbrand model has* $n + 1$ *elements.*

Hence, to decide whether $\mathsf{Ax} \cup \{F_1\} \models F_2$ *holds, it is sufficient to decide whether* $\mathsf{Ax} \wedge F_1 \wedge \neg F_2'[x_1 : c_1, \ldots, x_n := c_n]$ *has a model with at most* $n + 1$ *elements (and return the opposite).*

$\Diamond$