Logic and Proof

Supervision 2 – Solutions

5. Formal reasoning in first-order logic

1. a) For each of the following FOL formulas, circle the free variables, and underline the bound variables. Connect each bound variable to its binding occurrence (either graphically, by numbering, or whatever works for you).

   i. $\forall x.\ x = x$

   ii. $\exists x.\ P(x,y) \land \forall y.\ \neg P(y,x)$

   iii. $(\forall x.\ P(x,y)) \to (\exists y.\ P(x,y))$

   iv. $\exists z.\ P(x,y) \land Q(x,z)$

   v. $\forall x.\ (P(x) \to Q(x)) \land S(x,y)$

   vi. $P(x, \forall z.\ (\exists x.\ Q(y,x)) \to Q(x,z))$

   > The scope of the binding extends as far after the dot as possible, within the limits of any outside parentheses.

   i. $\forall x.\ \underline{x} = \underline{x}$

   ii. $\exists x.\ P(\underline{x}, \textcircled{y}) \land \forall y.\ \neg P(\underline{y}, \underline{x})$

   iii. $\Big(\forall x.\ P(\underline{x}, \textcircled{y})\Big) \to \Big(\exists y.\ P(\textcircled{x}, \underline{y})\Big)$

   iv. $\exists z.\ P(\textcircled{x}, \textcircled{y}) \land Q(\textcircled{x}, \underline{z})$

   v. $\forall x.\ (P(\underline{x}) \to Q(\underline{x})) \land S(\underline{x}, \textcircled{y})$

   vi. $P\Big(\textcircled{x}, \forall z.\ (\exists x.\ Q(\textcircled{y}, \underline{x})) \to Q(\textcircled{x}, \underline{z})\Big)$

   b) Apply the substitution $[x \mapsto f(x,y), y \mapsto g(z)]$ to the formulas above.

   > Substitution occurs only for *free* variables. Occasionally we need to α-rename the bound variables to avoid variable capture: free variables in terms to be substituted must remain free.
   >
   > - $\forall x.\ \underline{x} = \underline{x}$ (all variables are bound, so no substitution happens)
   > - $\exists x.\ P(\underline{x}, g(\textcircled{z})) \land \forall y.\ \neg P(\underline{y}, \underline{x})$
   > - $\Big(\forall x.\ P(\underline{x}, g(\textcircled{z}))\Big) \to \Big(\exists w.\ P(f(\textcircled{x}, \textcircled{y}), \underline{w})\Big)$ (y has to be renamed)
   > - $\exists w.\ P(f(\textcircled{x}, \textcircled{y}), g(\textcircled{z})) \land Q(f(\textcircled{x}, \textcircled{y}), \underline{w})$ (z has to be renamed)
   > - $\forall x.\ (P(\underline{x}) \to Q(\underline{x})) \land S(\underline{x}, g(\textcircled{z}))$
   > - $P\Big(f(\textcircled{x}, \textcircled{y}), \forall w.\ (\exists x.\ Q(g(\textcircled{z}), \underline{x})) \to Q(f(\textcircled{x}, \textcircled{y}), \underline{w})\Big)$ (z renamed)

2. Consider the following proof attempt of the set-theoretic conjecture:

$$\forall A, B.\ \ \mathcal{P}(A \cup B) = \mathcal{P}(A) \cup \mathcal{P}(B)$$

Proof. Let A, B be sets. We prove the proposition using equational reasoning:

$$\begin{aligned} \mathcal{P}(A \cup B) &= \{X \mid X \subseteq A \cup B\} && \text{(def. of powerset)} \\ &= \{X \mid \forall x.\ x \in X \to x \in A \cup B\} && \text{(def. of subsets)} \end{aligned}$$

$$
\begin{aligned}
&= \{X \mid \forall x.\, x \in X \to (x \in A \vee x \in B)\} && \text{(def. of union)}\\
&= \{X \mid \forall x.\, (x \in X \to x \in A) \vee (x \in X \to x \in B)\} && \text{(lemma of prop. logic)}\\
&= \{X \mid (X \subseteq A) \vee (X \subseteq B)\} && \text{(def. of subsets)}\\
&= \{X \mid X \subseteq A\} \cup \{X \mid X \subseteq B\} && \text{(def. of union)}\\
&= \mathcal{P}(A) \cup \mathcal{P}(B) && \text{(def. of powerset)}
\end{aligned}
$$

□

Is this a valid proof? Why or why not?

> This is not a valid proof, and no "proof" of it may be correct since the theorem itself is false. Most of the steps are actually correct, and the mistake is somewhat hidden. The problem is going from
>
> $$\forall x.\, (x \in X \to x \in A) \vee (x \in X \to x \in B) \tag{1}$$
>
> to
>
> $$(X \subseteq A) \vee (X \subseteq B) \tag{2}$$
>
> Even though the definition of the subset relation $S \subseteq A$ is $\forall x.\, x \in S \to x \in A$ and we have two similar implications in step (1), we would actually need an intermediate isomorphism
>
> $$(\forall x.\, x \in X \to x \in A) \vee (\forall x.\, x \in X \to x \in B)$$
>
> to be able to get to (2), from which the rest of the proof would follow. However, universal quantification *does not* distribute over disjunction, only conjunction:
>
> $$\forall x.\, P(x) \vee Q(x) \not\simeq (\forall x.\, P(x)) \vee (\forall y.\, Q(y))$$
>
> Dually, existential quantification distributes over disjunction, but does not distribute over conjunction. These properties follow from the interpretation of universal quantification as a generalised conjunction, and existential quantification as a generalised disjunction.

3. Verify the following equivalences by appealing to the truth definition of FOL.

$$\neg(\exists x.\, P(x)) \simeq \forall x.\, \neg P(x) \qquad (\forall x.\, P(x)) \wedge R \simeq \forall x.\, (P(x) \wedge R)$$
$$(\exists x.\, P(x)) \vee (\exists x.\, Q(x)) \simeq \exists x.\, (P(x) \vee Q(x))$$

> While semantic equivalences in propositional logic can be established by comparing truth tables (see Ex. 2.1), this is not possible in FOL due to the presence of quantifiers and variables. The associated notion of semantic truth is therefore more involved, making use of domains, interpretations, valuations, and a structural inductive definition of the truth of FOL formulae. But, in essence, we interpret a FOL formula as the intuitive English translation for what it says, and establish equivalences via this interpretation.
>
> Assume a domain D and interpretation I. Two formulae A and B are equivalent, if $\vDash_{\mathcal{I},V} A$ holds if and only if $\vDash_{\mathcal{I},V} B$ holds for some valuation V.
>
> - $\neg(\exists x.\, P(x)) \simeq \forall x.\, \neg P(x)$: The LHS holds if there does not exist an element $d \in D$

such that $I[P](d) = 1$. The RHS holds if for all $d \in D$, $I[P](d) = 1$ does not hold – that is, a d for which $I[P](d) = 1$ holds cannot exist, which is precisely the LHS.

- $(\forall x.\, P(x)) \wedge R \simeq \forall x.\, (P(x) \wedge R)$: Both sides are true if and only if $I[R] = 1$ and $I[P](d) = 1$ for all $d \in D$.
- $(\exists x.\, P(x)) \vee (\exists x.\, Q(x)) \simeq \exists x.\, (P(x) \vee Q(x))$: Both sides are true if and only if there exists an element $d \in D$ such that either $I[P](d) = 1$ or $I[Q](d) = 1$.

4. Prove the equivalence $(\forall x.\, P(x) \vee P(a)) \simeq P(a)$.

There are several approaches. We can use equivalence reasoning to show that

$$(\forall x.\, P(x) \vee P(a)) \leftrightarrow P(a)$$

is a theorem of FOL. Alternatively, we can make use of the fact that the sequent calculus is a sound and complete proof system for FOL, so we can also establish the sequents $(\forall x.\, P(x) \vee P(a)) \Rightarrow P(a)$ and $P(a) \Rightarrow (\forall x.\, P(x) \vee P(a))$. We can even mix the two techiques: rewrite the LHS using the FOL equivalence $\forall x.\, P(x) \vee P(a) \simeq (\forall x.\, P(x)) \vee P(a)$, then do two sequent proofs:

$$\dfrac{\dfrac{\dfrac{}{P(a) \Rightarrow P(a)}}{\forall x.\, P(x) \Rightarrow P(a)}\,(\forall l, a/x) \qquad \dfrac{}{P(a) \Rightarrow P(a)}}{(\forall x.\, P(x)) \vee P(a) \Rightarrow P(a)}\,(\vee l) \qquad \dfrac{\dfrac{}{P(a) \Rightarrow \forall x.\, P(x), P(a)}}{P(a) \Rightarrow (\forall x.\, P(x)) \vee P(a)}\,(\vee r)$$

5. Prove the following sequents. *Hint*: the last one requires two uses of $(\forall l)$.

$$(\forall x.\, P(x)) \wedge (\forall x.\, Q(x)) \Rightarrow \forall y.\, (P(y) \wedge Q(y))$$
$$\forall x.\, P(x) \wedge Q(x) \Rightarrow (\forall y.\, P(y)) \wedge (\forall y.\, Q(y))$$
$$\forall x.\, P(x) \to P(f(x)), P(a) \Rightarrow P(f(f(a)))$$

These proofs are largely straightforward. Some re-ordering of the steps is allowed, though not of the quantifier inferences.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{P(y), \forall x.\, Q(x) \Rightarrow P(y)}}{\forall x.\, P(x), \forall x.\, Q(x) \Rightarrow P(y)}\,(\forall l, y/x) \qquad \dfrac{\dfrac{}{\forall x.\, P(x), Q(y) \Rightarrow Q(y)}}{\forall x.\, P(x), \forall x.\, Q(x) \Rightarrow Q(y)}\,(\forall l, y/x)}{\forall x.\, P(x), \forall x.\, Q(x) \Rightarrow P(y) \wedge Q(y)}\,(\wedge r)}{\forall x.\, P(x), \forall x.\, Q(x) \Rightarrow \forall y.\, P(y) \wedge Q(y)}\,(\forall r)}{\forall x.\, P(x) \wedge \forall x.\, Q(x) \Rightarrow \forall y.\, P(y) \wedge Q(y)}\,(\wedge l)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P(y), Q(y) \Rightarrow P(y)}}{P(y) \wedge Q(y) \Rightarrow P(y)}\,(\wedge l)}{\forall x.\, P(x) \wedge Q(x) \Rightarrow P(y)}\,(\forall l, y/x)}{\forall x.\, P(x) \wedge Q(x) \Rightarrow \forall y.\, P(y)}\,(\forall r) \qquad \dfrac{\dfrac{\dfrac{\dfrac{}{P(y), Q(y) \Rightarrow Q(y)}}{P(y) \wedge Q(y) \Rightarrow Q(y)}\,(\wedge l)}{\forall x.\, P(x) \wedge Q(x) \Rightarrow Q(y)}\,(\forall l, y/x)}{\forall x.\, P(x) \wedge Q(x) \Rightarrow \forall y.\, Q(y)}\,(\forall r)}{\forall x.\, P(x) \wedge Q(x) \Rightarrow (\forall y.\, P(y)) \wedge (\forall y.\, Q(y))}\,(\wedge r)$$

We will write $f^2(a)$ for $f(f(a))$ and Γ for $\forall x.\, P(x) \to P(f(x))$ to save space, and also omit some parentheses.

$$\dfrac{\dfrac{}{P(a) \Rightarrow P(a), P(f^2a)} \qquad \dfrac{\dfrac{\dfrac{}{P(fa), P(a) \Rightarrow P(fa), P(f^2a)} \qquad \dfrac{}{P(f^2a), P(fa), P(a) \Rightarrow P(f^2a)}\,(\to l)}{P(fa) \to P(f^2a), P(fa), P(a) \Rightarrow P(f^2a)}\,(\forall l, fa/x)}{\forall x.\, P(x) \to P(fx), P(fa), P(a) \Rightarrow P(f^2a)}\,(\to l)}{\forall x.\, P(x) \to P(fx), P(a) \to P(fa), P(a) \Rightarrow P(f^2a)}\,(\forall l, a/x)$$
$$\forall x.\, P(x) \to P(fx), P(a) \Rightarrow P(f^2a)$$

6. Prove the following sequents. *Hint*: the last one requires two uses of $(\exists r)$.

$$P(a) \vee \exists x.\, P(f(x)) \Rightarrow \exists y.\, P(y)$$
$$\exists x.\, P(x) \vee Q(x) \Rightarrow (\exists y.\, P(y)) \vee (\exists y.\, Q(y))$$
$$\Rightarrow \exists z.\, P(z) \to P(a) \wedge P(b)$$

Once again, some reordering of steps is possible, but the quantifier steps must be done in the other shown since otherwise the constraint on $(\exists l)$ is likely to be violated.

$$\dfrac{\dfrac{\dfrac{}{P(a) \Rightarrow P(a)}}{P(a) \Rightarrow \exists y.\, P(y)}\,(\exists r, a/y) \qquad \dfrac{\dfrac{\dfrac{}{P(f(x)) \Rightarrow P(f(x))}}{P(f(x)) \Rightarrow \exists y.\, P(y)}\,(\exists r, f(x)/y)}{\exists x.\, P(f(x)) \Rightarrow \exists y.\, P(y)}\,(\exists l)}{P(a) \vee \exists P.\, (f(x)) \Rightarrow \exists y.\, P(y)}\,(\vee l)$$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{P(x) \Rightarrow P(x), \exists y.\, Q(y)}}{P(x) \Rightarrow \exists y.\, P(y), \exists y.\, Q(y)}\,(\exists r, x/y) \qquad \dfrac{\dfrac{\dfrac{}{Q(x) \Rightarrow \exists y.\, P(y), Q(x)}}{Q(x) \Rightarrow \exists y.\, P(y), \exists y.\, Q(y)}\,(\exists r, x/y)}{P(x) \vee Q(x) \Rightarrow \exists y.\, P(y), \exists y.\, Q(y)}\,(\vee l)}{P(x) \vee Q(x) \Rightarrow \exists y.\, P(y) \vee \exists y.\, Q(y)}\,(\vee r)}{\exists x.\, P(x) \vee Q(x) \Rightarrow (\exists y.\, P(y)) \vee (\exists y.\, Q(y))}\,(\exists l)$$

In the proof below, the right-hand side $P(a) \wedge P(b), P(a) \wedge P(b)$ can be collapsed to the single formula $P(a) \wedge P(b)$. This makes sense because we regard a sequent as a set of

formulas, and the transformation is sound due to the idempotence of disjunction.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{P(a), P(b) \Rightarrow P(a) \quad P(a), P(b) \Rightarrow P(b)}{P(a), P(b) \Rightarrow P(a) \wedge P(b), P(a) \wedge P(b)}\,(\wedge r)}{P(a) \Rightarrow P(a) \wedge P(b), P(b) \to P(a) \wedge P(b)}\,(\to r)}{P(a) \Rightarrow P(a) \wedge P(b), \exists z.\, P(z) \to P(a) \wedge P(b)}\,(\exists r, b/z)}{\Rightarrow P(a) \to P(a) \wedge P(b), \exists z.\, P(z) \to P(a) \wedge P(b)}\,(\to r)}{\Rightarrow \exists z.\, P(z) \to P(a) \wedge P(b)}\,(\exists r, a/z)$$

7. Prove the formula $\neg\forall y.\, (Q(a) \vee Q(b)) \wedge \neg Q(y)$ using equivalences, and then formally using the sequent calculus.

The semantic proof relies on standard FOL equivalences, including the *expansion* rules which let us extract instances of quantified formulae with the appropriate connective (conjunction for $\forall$, disjunction for $\exists$).

$$\begin{aligned}
&\neg\forall y.\, (Q(a) \vee Q(b)) \wedge \neg Q(y) \\
\simeq\ &\exists y.\, \neg((Q(a) \vee Q(b)) \wedge \neg Q(y)) && \text{(generalised de Morgan)} \\
\simeq\ &\exists y.\, \neg(Q(a) \vee Q(b)) \vee \neg\neg Q(y) && \text{(propositional de Morgan)} \\
\simeq\ &\exists y.\, \neg(Q(a) \vee Q(b)) \vee Q(y) && \text{(double negation elimination)} \\
\simeq\ &\neg(Q(a) \vee Q(b)) \vee \exists y.\, Q(y) && \text{(reduce scope of existential)} \\
\simeq\ &\neg(Q(a) \vee Q(b)) \vee Q(a) \vee Q(b) \vee \exists y.\, Q(y) && \text{(expansion twice)} \\
\simeq\ &\top \vee \exists y.\, Q(y) && \text{(law of excluded middle)} \\
\simeq\ &\top && \text{(truth is annihilator for disjunction)}
\end{aligned}$$

The sequent calculus proof is as follows:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{Q(a), \neg Q(b) \Rightarrow Q(a)}{Q(a), \neg Q(b), \neg Q(a) \Rightarrow}\,(\neg l) \quad \dfrac{Q(b), \neg Q(a) \Rightarrow Q(b)}{Q(b), \neg Q(a), \neg Q(b) \Rightarrow}\,(\neg l)}{Q(a) \vee Q(b), \neg Q(a), \neg Q(b) \Rightarrow}\,(\vee l)}{Q(a) \vee Q(b), \neg Q(a), (Q(a) \vee Q(b)) \wedge \neg Q(b) \Rightarrow}\,(\wedge l)}{(Q(a) \vee Q(b)) \wedge \neg Q(a), (Q(a) \vee Q(b)) \wedge \neg Q(b) \Rightarrow}\,(\wedge l)}{(Q(a) \vee Q(b)) \wedge \neg Q(a), \forall y.\, (Q(a) \vee Q(b)) \wedge \neg Q(y) \Rightarrow}\,(\forall l, b/y)}{\forall y.\, (Q(a) \vee Q(b)) \wedge \neg Q(y) \Rightarrow}\,(\forall l, a/y)}{\Rightarrow \neg\forall y.\, (Q(a) \vee Q(b)) \wedge \neg Q(y)}\,(\neg r)$$

6. Clause methods for propositional logic

1. Outline the steps of the Davis–Putnam–Logeman–Loveland method. Explain the goal of the method, and why the steps of the algorithm are sound. Why does the empty clause represent a contradiction?

DPLL is an algorithm for deciding the satisfiability of propositional formulae in clausal (conjunctive normal) form. Given a formula presented as a set of clauses, DPLL can either determine that the formula is not satisfiable, or return a satisfying model if it is. DPLL can also be used to determine if a formula is valid (i.e. satisfied by every interpretation) by negating it first and trying to derive a contradiction: if DPLL reaches the contradiction, there does not exist a falsifying interpretation of the original formula, so it must be valid. The contradiction is represented by the *empty clause* $\{\ \}$: since a clause $\{P_1, \ldots, P_n\}$ stands for a disjunction $P_1 \vee \cdots \vee P_n$, and the disjunction of no formulae is false, deriving the empty clause is equivalent to deriving falsity from the clauses.

The steps of the DPLL algorithm to establish the validity of a formula φ are as follows.

a) Negate φ and convert it to conjunctive normal form, then clauses.

b) Delete *tautological clauses* $\{P, \neg P, \ldots\}$. These are true for any interpretation of P, since the conjunct will contain $P \vee \neg P \simeq \top$.

c) *Unit propagation*: for every *unit clause* containing a single literal $\{L\}$ (where $L = P$ or $L = \neg P$ for some P), delete every clause containing L, and delete $\neg L$ from every remaining clause. This is sound since any model of the clauses must assign true to L (otherwise the corresponding conjunct would be false); any clause containing L will therefore be true, and the negation of L (which will be assigned $\bot$) will be a neutral element in the remaining clauses.

d) *Pure literal elimination*: remove every clause containing a *pure literal*, i.e. a literal L for which there is no clause containing $\neg L$. This is sound since we can always assign pure literals to be true (it cannot lead to a contradiction, since there is no clause containing their negation), so all clauses containing them will be automatically satisfied.

e) If an empty clause is reached, we reached a contradiction. Conversely, if all clauses are deleted, the original clause set is satisfiable, and the model can be determined from the assignments performed in the unit propagation and pure literal elim. steps.

f) Otherwise (no unit clauses, no pure literals, nonempty clauses left), choose a literal L to case-split on, and recursively apply the algorithm to the L and $\neg L$ subcases. The clause set is satisfiable if and only if one of the subcases is satisfiable. Since we exhaustively check both cases, this step is also sound.

2. Apply the DPLL procedure to the clause set:

$$\{P, Q\} \qquad \{\neg P, Q\} \qquad \{P, \neg Q\} \qquad \{\neg P, \neg Q\}$$

We cannot perform unit propagation or pure literal elimination, so we need to start with a case-split. Choosing P, we get:

- If P is true, we can unit-propagate it to get the clauses $\{Q\}, \{\neg Q\}$ which give the empty clause by unit-propagating Q.

- If P is false, we again get the clauses $\{ Q \}, \{ \neg Q \}$ and derive a contradiction.

3. Explain the resolution algorithm and how it differs from DPLL.

   The *resolution rule* is a rule inference that gives rise to a refutation theorem-proving algorithm for propositional (and first-order) logic. The clausal resolution step combines two clauses containing *complementary literals*: the clauses $\{ \Gamma, P \}$ and $\{ \neg P, \Delta \}$ can be resolved into the single clause $\{ \Gamma, \Delta \}$, where Γ and Δ stand for an arbitrary sequence of literals. The intuition behind the resolution step is nothing more than gluing together implications: the clause $\{ \Gamma, P \}$ is equivalent to $\neg\Gamma \to P$, $\{ \neg P, \Delta \}$ is equivalent to $P \to \Delta$, and the resolution step corresponds to the conclusion $\neg\Gamma \to \Delta$ which follows from the transitivity of $\to$.

   $$\frac{\{A_1, \ldots, A_m, B\} \qquad \{\neg B, C_1, \ldots, C_n\}}{\{A_1, \ldots, A_m, C_1, \ldots, C_n\}} \text{ (res B)}$$

   The resolution algorithm is based on proof by contradiction, so the original formula has to be negated before converting into clausal form. The rule is repeatedly applied to the clauses until the empty clause is reached; if we haven't reached the empty clause and cannot apply any more resolution steps, the original set of formulae is satisfiable. Resolution is *complete*, so if a set of clauses is unsatisfiable, resolution will be able to derive a contradiction (i.e. it won't get "stuck").

   DPLL and resolution are not interchangeable and should not be mix-and-matched. DPLL *modifies* and *consumes* clauses: unit propagation and pure literal elimination removes literals from clauses and deletes the clauses themselves, until the empty clause is created or all clauses are deleted. Resolution simply *combines* clauses by making transitive inferences and adding the resolvent to the clause set: the existing clauses are not modified or removed, and one clause can be used several times. DPLL can be used both to find models of a clause set (see SMT solvers), or perform refutation proofs of validity. Resolution is tailored to refutation proofs: getting "stuck" in a resolution proof does not give us a model, and we can never derive the empty clause set, since resolution does not decrease the number of clauses. The main advantage of resolution is that it is very simple to automate and can be made efficient with heuristics for choosing the pair of clauses to resolve.

4. Use resolution (showing the steps of converting the formula into clauses) to prove at least three of the following formulas.

$$(P \to Q \lor R) \to ((P \to Q) \lor (P \to R))$$
$$((P \to Q) \to P) \to P$$
$$(Q \to R) \land (R \to P \land Q) \land (P \to Q \lor R) \to (P \leftrightarrow Q)$$
$$(P \land Q \to R) \land (P \lor Q \lor R) \to ((P \leftrightarrow Q) \to R)$$
$$(P \to R) \land (R \land P \to S) \to (P \land Q \to R \land S)$$

a) $(P \to Q \vee R) \to ((P \to Q) \vee (P \to R))$

When negating a formula, it's useful to remember some equivalences to avoid having to do a lot of unnecessary work. The most useful one is $\neg(P \to Q) \simeq P \wedge \neg Q$, so when negating any implication, the hypothesis can be immediately read off as a clause (without negation, but perhaps with some rearranging). Thus, in the first formula, $(P \to Q \vee R)$ gives one of our clauses, then $\neg((P \to Q) \vee (P \to R)) \simeq (P \wedge \neg Q) \wedge (P \wedge \neg R)$ gives the rest:

$$① \{\neg P, Q, R\} \quad ② \{P\} \quad ③ \{\neg Q\} \quad ④ \{\neg R\}$$

$$① + ② \overset{P}{\Rightarrow} ⑤ \{Q, R\} \qquad ⑤ + ③ \overset{Q}{\Rightarrow} ⑥ \{R\} \qquad ⑥ + ④ \overset{R}{\Rightarrow} \square$$

b) $((P \to Q) \to P) \to P$

Negate and convert to clauses:

$$\begin{aligned}\neg((P \to Q) \to P) \to P &\simeq ((P \to Q) \to P) \wedge \neg P \simeq (\neg(P \to Q) \vee P) \wedge \neg P \\ &\simeq ((P \wedge \neg Q) \vee P) \wedge \neg P \simeq (P \vee P) \wedge (\neg Q \vee P) \wedge \neg P\end{aligned}$$

$$① \{P\} \quad ② \{\neg Q, P\} \quad ③ \{\neg P\}$$

A single resolution step on ① and ③ derives $\square$, so the original formula (Peirce's law) is valid.

c) $(Q \to R) \wedge (R \to P \wedge Q) \wedge (P \to Q \vee R) \to (P \leftrightarrow Q)$

Negate and convert to clauses. Note how negation only affects the consequent $P \leftrightarrow Q$; the hypotheses can readily be read off as clauses:

- $Q \to R$ gives $\{\neg Q, R\}$
- $R \to (P \wedge Q) \simeq (R \to P) \wedge (R \to Q)$ gives $\{\neg R, P\}$ and $\{\neg R, Q\}$
- $P \to Q \vee R$ gives $\{\neg P, Q, R\}$

For the negated conclusion, convert $\neg(P \leftrightarrow Q)$ into clauses:

$$\neg(P \leftrightarrow Q) \simeq P \leftrightarrow \neg Q \simeq (P \to \neg Q) \wedge (\neg Q \to P) \simeq (\neg P \vee \neg Q) \wedge (Q \vee P)$$

$$① \{\neg Q, R\} \quad ② \{\neg R, P\} \quad ③ \{\neg R, Q\} \quad ④ \{\neg P, Q, R\} \quad ⑤ \{\neg P, \neg Q\} \quad ⑥ \{P, Q\}$$

One possible sequence of resolution steps is below. This is an example of *linear resolution*, where the output of one resolution step is the input to the next one – it can also be represented as a left-branching tree.

$$\begin{aligned}① + ② \overset{R}{\Rightarrow} \{\neg Q, P\} + ⑥ \overset{Q}{\Rightarrow} \{P\} + ④ \overset{P}{\Rightarrow} \{Q, R\} \\ + ③ \overset{R}{\Rightarrow} \{Q\} + ⑤ \overset{Q}{\Rightarrow} \{\neg P\} + \{P\} \overset{P}{\Rightarrow} \square\end{aligned}$$

d) $(P \wedge Q \to R) \wedge (P \vee Q \vee R) \to ((P \leftrightarrow Q) \to R)$

Negate and convert to clauses. Again, a lot of the formula goes untouched; we read off the following hypotheses:

- $P \wedge Q \to R$ gives $\{\neg P, \neg Q, R\}$
- $P \vee Q \vee R$ gives $\{P, Q, R\}$
- $P \leftrightarrow Q \simeq (P \to Q) \wedge (Q \to P)$ gives $\{\neg P, Q\}, \{P, \neg Q\}$

Finally, with the negated conclusion $\neg R$, the clause set is:

$$① \{\neg P, \neg Q, R\} \quad ② \{P, Q, R\} \quad ③ \{\neg P, Q\} \quad ④ \{P, \neg Q\} \quad ⑤ \{\neg R\}$$

Below is one possible resolution sequence.

$$① + ⑤ \overset{R}{\Rightarrow} ⑥ \{\neg P, \neg Q\} \qquad ② + ⑤ \overset{R}{\Rightarrow} ⑦ \{P, Q\}$$

$$⑥ + ④ \overset{P}{\Rightarrow} ⑧ \{\neg Q\} \qquad ⑦ + ③ \overset{P}{\Rightarrow} ⑨ \{Q\} \qquad ⑧ + ⑨ \overset{Q}{\Rightarrow} \square$$

The last few steps were analogous to the second example in Section 6.4 of the notes. Collapsing duplicate clauses like $\{Q, Q\}$ is allowed and often necessary.

e) $(P \to R) \wedge (R \wedge P \to S) \to (P \wedge Q \to R \wedge S)$

Negate and convert to clauses. Only $R \wedge S$ gets negated, the rest of the clauses can be read off from the hypotheses:

- $P \to R$ gives $\{\neg P, R\}$
- $R \wedge P \to S$ gives $\{\neg R, \neg P, S\}$
- $P \wedge Q$ gives $\{P\}, \{Q\}$

With the negated conclusion $\neg R \vee \neg S$, the clause set is:

$$① \{\neg P, R\} \quad ② \{\neg P, \neg R, S\} \quad ③ \{P\} \quad ④ \{Q\} \quad ⑤ \{\neg R, \neg S\}$$

Below is a possible linear resolution sequence.

$$① + ② \overset{R}{\Rightarrow} \{\neg P, S\} + ⑤ \overset{S}{\Rightarrow} \{\neg P, \neg R\} + ① \overset{R}{\Rightarrow} \{\neg P\} + ③ \overset{P}{\Rightarrow} \square$$

5. Convert these axioms to clauses, showing all steps. Then prove $\text{Winterstorm} \to \text{Miserable}$ by resolution.

$$\text{Wet} \wedge \text{Cold} \to \text{Miserable}$$
$$\text{Winterstorm} \to \text{Storm} \wedge \text{Cold}$$
$$\text{Storm} \to \text{Rain} \wedge \text{Windy}$$
$$\text{Rain} \wedge (\text{Windy} \vee \neg\text{Umbrella}) \to \text{Wet}$$

The propositions listed are our *knowledge base*: the set of axioms that we assume to be true. Our task is to prove $\text{Winterstorm} \to \text{Miserable}$ using resolution, i.e. that this implication

follows from the axioms in our knowledge base. Resolution involves negating the proof goal, since it proceeds by refuting the negation of the proposition; importantly, our initial axioms are *not* negated, as they are assumptions, not goals. Another way to see this is to call the conjunction of the axioms Γ, and the proof goal $\text{Winterstorm} \to \text{Miserable}\ P$. We are required to prove the proposition $\Gamma \to P$, i.e. P follows from the axioms Γ. To do a resolution proof, we negate this whole implication, and as seen before, this leaves the hypotheses Γ untouched, only negating P: $\neg(\Gamma \to P) \simeq \Gamma \wedge \neg P$. Hence, the axioms are converted to clauses without negation, then we add the clauses we get from negating $\text{Winterstorm} \to \text{Miserable}$ and do resolution on the whole clause set.

With some practice, converting formulae into clauses "on the fly" becomes easy; it's important to remember the standard propositional equivalences, and whether the formulae are negated or not! Below are some of the initial steps:

- $\text{Wet} \wedge \text{Cold} \to \text{Mis} \simeq \neg\text{Wet} \vee \neg\text{Cold} \vee \text{Mis}$
- $\text{Winter} \to \text{Storm} \wedge \text{Cold} \simeq (\text{Winter} \to \text{Storm}) \wedge (\text{Winter} \to \text{Cold})$
- $\text{Storm} \to \text{Rain} \wedge \text{Wind} \simeq (\text{Storm} \to \text{Rain}) \wedge (\text{Storm} \to \text{Wind})$
- $\text{Rain} \wedge (\text{Wind} \vee \neg\text{Umb}) \to \text{Wet} \simeq (\text{Rain} \wedge \text{Wind}) \vee (\text{Rain} \wedge \neg\text{Umb}) \to \text{Wet} \simeq (\text{Rain} \wedge \text{Wind} \to \text{Wet}) \wedge (\text{Rain} \wedge \neg\text{Umb} \to \text{Wet})$

We also get two clauses from $\neg(\text{Winter} \to \text{Mis})$, giving us the final clause set of:

$$① \{\neg\text{Wet}, \neg\text{Cold}, \text{Mis}\} \quad ② \{\neg\text{Winter}, \text{Storm}\} \quad ③ \{\neg\text{Winter}, \text{Cold}\}$$
$$④ \{\neg\text{Storm}, \text{Rain}\} \quad ⑤ \{\neg\text{Storm}, \text{Wind}\} \quad ⑥ \{\neg\text{Rain}, \neg\text{Wind}, \text{Wet}\}$$
$$⑦ \{\neg\text{Rain}, \text{Umb}, \text{Wet}\} \quad ⑧ \{\text{Winter}\} \quad ⑨ \{\neg\text{Mis}\}$$

One sequence of resolution steps is below. Note how we make use of unit clauses as much as possible – resolution with a unit clause is the only way to decrease the size of the clause, as the resolvent of clauses C and D will have size $|C| + |D| - 2$. Unit clauses can also be resolved "in bulk" with another clause containing the negation of the unit literals, as shown in the last step (and the elimination of clause ⑥ beforehand).

$$② + ⑧ \Longrightarrow Ⓐ \{\text{Storm}\} \quad ③ + ⑧ \Longrightarrow Ⓑ \{\text{Cold}\} \quad Ⓐ + ④ \Longrightarrow Ⓒ \{\text{Rain}\}$$
$$Ⓐ + ⑤ \Longrightarrow Ⓓ \{\text{Wind}\} \quad Ⓒ + ⑥ \Longrightarrow Ⓔ \{\neg\text{Wind}, \text{Wet}\} \quad Ⓓ + Ⓔ \Longrightarrow Ⓕ \{\text{Wet}\}$$
$$Ⓕ \{\text{Wet}\} + Ⓑ \{\text{Cold}\} + ⑨ \{\neg\text{Mis}\} + ① \{\neg\text{Wet}, \neg\text{Cold}, \text{Mis}\} \Longrightarrow \square$$

7. Skolem functions, Herbrand's Theorem and unification

1. a) Explain the process of Skolemisation on a formula of your choice.

Skolemisation is the transformation of first-order formulae that removes existential quantifiers while maintaining the consistency of the formula. Every existentially quantified variable is replaced by a new function symbol applied to all universally quantified variables bound outside of the scope of the existential. For example,

consider the FOL formula:

$$\Big(\forall x.\ \exists y.\ \forall z.\ (P(\underline{x}) \wedge Q(\underline{z}, \underline{y})) \vee (\exists w.\ R(\underline{w}))\Big) \leftrightarrow \exists y.\ \forall v.\ S(\underline{y}, \underline{v})$$

The variables y, w and y are existentially quantified and will be Skolemised to constant symbols of different arities:

- y variable gets bound in the scope of one universal quantification, so it gets Skolemised to $f(x)$
- w is also bound in the scope of z, so it gets replaced with $g(x, z)$
- y is different from the y on the LHS of $\leftrightarrow$ so it gets its own Skolem symbol. In this case, the binding is top-level, so the function has arity zero – that is, it is a constant value c.

Thus, the Skolemised formula is as follows:

$$\Big(\forall x.\ \forall z.\ (P(\underline{x}) \wedge Q(\underline{z}, f(x))) \vee R(g(x, z))\Big) \leftrightarrow \forall v.\ S(c, \underline{v})$$

b) The notes state that "[Skolemisation] does not preserve the meaning of a formula. However, it does preserve *inconsistency*, which is the critical property". Justify the two claims in this statement, demonstrating them on your example above.

The justification for Skolemisation can be seen by considering the Tarskian truth definition of a formula $\exists x.\ P(x)$. We say that $\exists x.\ P(x)$ is satisfiable (or consistent) if there exists an interpretation $\mathcal{I} = (I, D)$ such that for all valuations V, we have $\vDash_{\mathcal{I},V} \exists x.\ P(x)$. By the truth definition, this holds if there exists an $m \in D$ such that $\vDash_{\mathcal{I},V\{m/x\}} P(x)$ holds, which, in turn, is the case if $I[P]((V\{m/x\})(x)) = I[P](m) = 1$.

$$\exists x.\ P(x) \text{ satisfiable} \iff \exists I, D.\ \exists m \in D.\ I[P](m) = 1$$

The Skolemised version of the formula is simply $P(c)$ for some new constant symbol c, and $P(c)$ is consistent if there exists an interpretation $\mathcal{I}' = (I', D')$ such that for all valuations V' we have $\vDash_{\mathcal{I}',V'} P(c)$. By the truth definition, this holds if $I'[P](I'[c]) = 1$. The crucial point is that the only way $I'[c]$ can be defined is if there is an interpretation of the constant c in the domain D', i.e. an element $m \in D'$ such that $I'[c] = m$. Then:

$$P(c) \text{ satisfiable} \iff \exists I', D'.\ \exists m \in D'.\ I'[c] = m \wedge I[P](m) = 1$$

which is nothing more than the satisfiability condition of $\exists x.\ P(x)$. In short, if there is an interpretation of $\exists x.\ P(x)$, there must exist an interpretation of $P(c)$, and, contrapositively, if $P(c)$ is inconsistent then so is $\exists x.\ P(x)$. This reasoning can be extended to Skolem functions incorporating a sequence $\vec{x}$ of quantified variables in $\forall \vec{x}.\ \exists y.\ P(\vec{x}, y)$: if for all sequences of elements $\vec{n} \in D$ corresponding to $\vec{x}$ there must exist an $m \in D$ such that $I[P](\vec{n}, m)$ holds, we can construct a model for $\forall \vec{x}.\ P(f(\vec{x}))$ in which the interpretation of the function symbol f maps $\vec{n}$ to m as prescribed by the model for the original formula. If we can find no model satisfying $\forall \vec{x}.\ P(f(\vec{x}))$,

> we won't be able to find a model satisfying $\forall \vec{x}.\ \exists y.\ P(\vec{x}, y)$ either.
>
> For a concrete example consider the formula $\forall x.\ \exists y.\ P(x, y)$ (which states that P is a total relation) and take $P(x, y)$ to mean "y is a prime number greater than x", where x and y range over the naturals. By Euclid's proof of the infinitude of primes, we know that this is a satisfying interpretation: for any $x \in \mathbb{N}$, there exists a prime $y \in \mathbb{N}$ such that $x < y$. If there is an assignment of a prime y to every x in the domain $\mathbb{N}$, there must exist (at least one) function $f : \mathbb{N} \to \mathbb{N}$ that assigns x to a larger prime y, for example, the next largest one. This interpretation then satisfies the Skolemised formula $\forall x.\ P(x, f(x))$.
>
> Now, take the formula $\psi = \exists y.\ P(y) \wedge \neg P(y)$. Skolemising this gives $P(c) \wedge \neg P(c)$. It's easy to see that no interpretation for P or c exists that would satisfy this formula, which means ψ is unsatisfiable as well – there can't exist a y such that $P(y) \wedge \neg P(y)$.

2. Skolemize the following formulas, dropping all quantifiers.

$$\forall u.\ \exists x, y.\ P(x, y) \qquad \exists x, y.\ \forall z.\ \exists w.\ P(x, y, z, w) \qquad \forall u.\ (\exists x.\ P(x, x)) \wedge \forall v.\ \exists y.\ Q(u, y)$$

> Nothing too surprising here.
>
> $$P(f(u), g(u)) \qquad P(a, b, z, f(z)) \qquad P(f(u), f(u)) \wedge Q(u, g(u, v))$$

3. Consider a first-order language A with z and o as constant symbols, with n as a 1-place function symbol and a as a 2-place function symbol, and with C as a 2-place predicate symbol.

   a) Describe the Herbrand universe for this language.

   > The Herbrand universe of a first-order language is the set of all possible *closed* terms that can be constructed from the constants and function symbols of the language. It is an infinite set constructed recursively: taking any function symbol such as a, the Herbrand universe H must contain the terms $a(t_1, t_2)$ for all terms t_1 and t_2 taken from H itself. Some elements of H are:
   >
   > $$\{z, o, n(z), n(o), a(z, z), a(z, o), a(o, z), n(n(z)), n(n(o)), n(a(z, z)), a(n(o), z), \dots\}$$

   b) This language has an interpretation $\mathcal{I}$ in the domain $D = \mathbb{Z}$ of integers, with z and o interpreted as $0 \in \mathbb{Z}$ and $1 \in \mathbb{Z}$ respectively, n being the negation function $x \mapsto -x$, a being the addition function $x, y \mapsto x + y$, and C being the less-than comparison relation $x, y \mapsto x < y$. What is the Herbrand model of the symbols of the language with respect to this interpretation $\mathcal{I}$?

   > The Herbrand interpretation of a first-order language L with respect to a given model $\mathcal{I} = (D, I)$ is the "syntactic interpretation" of L which can be directly translated into $\mathcal{I}$. It is an example of a so-called *free construction* in mathematics, because the interpretation can be freely generated from the syntax; any real interpretation would involve some "creativity" (such as interpreting $a(o, z)$ and $a(z, o)$ as the same

number 1), while the Herbrand model simply interprets a term as itself, not bothering about actually giving it a meaning. The only place where we need to refer to the real interpretation is giving a meaning to relation symbols, which need to return a truth value even in the free interpretation.

In our example, the constant and function symbols are z, o, n and a; these must be interpreted as constants and functions on the desired domain, which, in this case, is the Herbrand universe H.

- Constant terms are interpreted as themselves: $I_H[z] = z \in H$, $I_H[o] = o \in H$
- $I_H[n] \colon H \to H$ is the function that takes a term $t \in H$, and prepends it with the symbol n (notice how we're not talking about "negating" the element – that happens in the standard interpretation in $\mathbb{Z}$). That is, $I_H[n](t) = n(t)$.
- $I_H[a] \colon H \times H \to H$ is the function that takes terms t_1 and t_2 in H, and combines them with the symbol a. You can think of this as creating a new syntax tree with root a and subtrees t_1 and t_2. That is, $I_H[a](t_1, t_2) = a(t_1, t_2)$.

The relation symbol < cannot be given such a trivial interpretation, which has to be a relation $I_H[<] \colon H \times H \to \mathbb{B}$ – the output must be an actual truth value, not just a syntax tree. Without having a real interpretation in $\mathbb{Z}$ to refer to, we cannot make any obvious choices on how to make sense of a formula like $\varphi = C(a(o, z), n(n(o)))$. Thus, we "ask" what an intended interpretation would say to this, and make that the Herbrand interpretation as well. The choice of interpretation matters: φ would be true if $I_{\mathbb{Z}}[C](x, y) = x \leq y$, but false if $I_{\mathbb{Z}}[C](x, y) = x < y$.

Formally, the relation $I_H[C] \colon H \times H \to \mathbb{B}$ takes two terms $t_1, t_2 \in H$, interprets them using our base model $I_{\mathbb{Z}}$, and compares the answers using the base interpretation of C: $I_H[C](t_1, t_2)$ iff $I_{\mathbb{Z}}(t_1) < I_{\mathbb{Z}}(t_2)$.

Herbrand interpretations are useful because every real interpretation of a set of first-order clauses *factorises* through a Herbrand interpretation in a unique way. That is, given an interpretation $\mathcal{I} = (D, I)$ of a set of clauses S (or a quantifier-free formula that may contain variables), there exists a unique function $\hat{I} \colon H \to D$ such that the interpretation $I[t]$ of a term coincides with $\hat{I}(I_H[t])$ for all terms t. Instead of interpreting the set of clauses directly, we can take a "detour" through the systematically generated Herbrand interpretation in a consistency-preserving way.

For example, take the clauses

$$S = \{\, C(x, a(x, o)) \,\} \quad \{\, C(x, x), C(z, o) \,\}$$

These stand for the FOL formula $(\forall x.\ C(x, a(x, o))) \wedge (\forall y.\ C(y, y) \vee C(z, o))$. We have the above interpretation $(\mathbb{Z}, I_{\mathbb{Z}})$ which satisfies S: it is indeed the case that $x < x + 1$ for all $x \in \mathbb{Z}$, and that either $y < y$ or $0 < 1$ holds for all y. The claim of Lemma 12 in the notes is that there must exist a Herbrand interpretation satisfying

S. The universe H is as described above, consisting of elements such as $a(n(o), z)$ and $n(n(n(z)))$. Similarly, interpretations of the constant and functions symbols are the identity: $I_H[a(n(o), z)] = a(n(o), z)$. The interpretation of C is where we need to make use of $I_{\mathbb{Z}}$, as explained above. To show that I_H satisfies the clauses, we need to prove that $I_H[C](t, a(t, o))$ holds for all terms $t \in H$ (and similarly for the second clause). How do we know that this is true? Well, $I_H[C](t, a(t, o)) \leftrightarrow I_{\mathbb{Z}}[t] < I_{\mathbb{Z}}[a(t, o)] \leftrightarrow I_{\mathbb{Z}}[t] < I_{\mathbb{Z}}[t] + 1$ for all $t \in H$, but this certainly holds, since we know that $x < x + 1$ for all $x \in \mathbb{Z}$ – in particular, $I_{\mathbb{Z}}[t]$. The real interpretation $I_{\mathbb{Z}}$ can be split into the "freely generated" Herbrand interpretation I_H, followed by a unique function $\hat{I}_{\mathbb{Z}} : H \to \mathbb{Z}$, mapping the ground terms of the Herbrand universe to integers.

4. For at least three of the following pairs of terms, give a most general unifier or explain why none exists. Do not rename variables prior to performing the unification.

$f(g(x), z)$	$f(y, h(y))$
$j(x, y, z)$	$j(f(y, y), f(z, z), f(a, a))$
$j(x, z, x)$	$j(y, f(y), z)$
$j(f(x), y, a)$	$j(y, z, z)$
$j(g(x), a, y)$	$j(z, x, f(z, z))$

Things to pay attention to: occurs-check violations (e.g. trying to unify $f(x)$ with x), and not unifying symbols with different symbols. Substitutions also need to be accumulated when unifying arguments of an n-ary function symbol in sequence, and the final substitution is the composition of the component substitutions (see Section 7.6 of the notes).

- The MGU of $f(g(x), z)$ and $f(y, h(y))$ is $[g(x)/y, h(g(x))/z]$, and the common instance is $f(g(x), h(g(x)))$. As noted above, substitutions accumulate: in the second step we are unifying $z[g(x)/y] = z$ and $h(y)[g(x)/y] = h(g(x))$, not just z and $h(y)$. Without accumulation, the substitution would be $\sigma = [g(x)/y, h(y)/z]$ – but this is not even a unifier, because $f(g(x), z)[\sigma] = f(g(x), h(y))$ but $f(y, h(y))[\sigma] = f(g(x), h(g(x)))$.
- The MGU of $j(x, y, z)$ and $j(f(y, y), f(z, z), f(a, a))$ is the composition of $[f(y, y)/x]$, $[f(z, z)/y]$ and $[f(a, a)/z]$, namely:

$$[f(f(f(a, a), f(a, a)), f(f(a, a), f(a, a)))/x,\ f(f(a, a), f(a, a))/y,\ f(a, a)/z]$$

  This is a slightly contrived example to demonstrate how the naive unification algorithm can take exponential time.
- The terms $j(x, z, x)$ and $j(y, f(y), z)$ are not unifiable. A unifier must identify the variables x, y and z, and thus also unify y with $f(y)$, which violates the occurs check.

- The terms $j(f(x), y, a)$ and $j(y, z, z)$ are also not unifiable. We have to unify y both with a and $f(x)$, but a and $f(x)$ are not unifiable as they are different function symbols.
- The MGU of $j(g(x), a, y)$ and $j(z, x, f(z, z))$ is the composition of $[g(x)/z]$, $[a/x]$ and $[f(z, z)/y]$, namely $[a/x, f(g(a), g(a))/y, g(z)/z]$. The common instance is $j(g(a), a, f(g(a), g(a)))$.

5. Which of the following substitutions are most general unifiers for the terms $f(x, y, z)$ and $f(w, w, v)$?

$$[x/y, x/w, v/z] \qquad [y/x, y/w, v/z] \qquad [y/x, v/z]$$
$$[x/y, x/z, x/w, x/v] \qquad [u/x, u/y, u/w, y/z, y/v]$$

- $[x/y, x/w, v/z]$: this is a unifier with the common instance $f(x, x, v)$.
- $[y/x, y/w, v/z]$: this is a unifier with the common instance $f(y, y, v)$.
- $[y/x, v/z]$: this is not a unifier, as the first term becomes $f(y, y, v)$, while the other is $f(w, w, v)$. We're missing $[y/w]$, which the previous substitution had.
- $[x/y, x/z, x/w, x/v]$: this is a unifier with the common instance $f(x, x, x)$. But it unifies "more things" than required, that is, it's not a most general unifier: we can get to the same term by applying the first substitution $[x/y, x/w, v/z]$ above to get $f(x, x, v)$, then also substituting x for v. Thus, $[x/y, x/z, x/w, x/v] = [x/y, x/w, v/z] \circ [x/v]$, so this is not an MGU.
- $[u/x, u/y, u/w, y/z, y/v]$: this is a unifier with the common instance $f(u, u, y)$. Again, it performs more substitutions than required and can be decomposed (for example) as $[u/x, u/y, u/w, y/z, y/v] = [x/y, x/w, v/z] \circ [u/x, y/v]$.

8. First-order resolution

1. What techniques allow us to convert first-order formulas into "propositional" clauses, and prove them using resolution? How are quantifiers and variables handled?

See the *First-order resolution* supplement.

2. Is the clause $\{P(x, b), P(a, y)\}$ logically equivalent to the unit clause $\{P(a, b)\}$? Is the clause $\{P(y, y), P(y, a)\}$ logically equivalent to $\{P(y, a)\}$? Explain both answers.

Logical equivalence would imply that $(\forall xy.\ P(x, b) \lor P(a, y)) \leftrightarrow P(a, b)$ is valid. Of course, this is not the case: $P(a, b) \to (\forall xy.\ P(x, b) \lor P(a, y))$ is clearly wrong. We can find a falsifying model over the domain $\{0, 1\}$: $0 < 1 \not\to (1 < 1 \lor 0 < 0)$.

The second pair of clauses is also not equivalent: $\forall y.\ P(y, y) \lor P(y, a)$ does not imply $\forall x.\ P(x, a)$ because $\forall y.\ P(y, y) \to P(y, a)$ is not valid. Again, a falsifying model over $\{0, 1\}$ could be: $0 = 0 \not\to 0 = 1$.

> This shows that factoring usually results in logically weaker clauses so it's worth retaining the original clause if we want our proof procedure to be complete.

3. Show that every set S of definite clauses is consistent. *Hint*: first consider propositional logic, then extend your argument to first order logic.

> Definite clauses contain exactly one positive literal: they are of the form $\{\neg A_1, \ldots, \neg A_n, B\}$ and can be interpreted as implications $A_1 \wedge \cdots \wedge A_n \to B$. To satisfy a set of these implications, it is sufficient to ignore the hypotheses and satisfy the consequent (since $A \to \top$ is a tautology). Thus, all we need to do is set the positive literals occurring in each clause to true, and by the definite clause guarantee every clause will be satisfied. The same idea works for first-order clauses, except the positive literal may contain variables: these will need to be universally quantified. For instance, $A_1(x) \wedge \cdots \wedge A_n(x) \to B(x)$ is satisfied by the interpretation $B(x) = \top$ for all $x \in D$. Note that we can't use the Herbrand logic here: satisfying a single ground instance of a clause does not satisfy the full clause.

4. Convert the following formulas into clauses, showing each step: negating the formula, eliminating $\to$ and $\leftrightarrow$, pushing in negations, Skolemising, dropping the universal quantifiers, and converting the resulting formula into CNF. Apply resolution (and possibly factoring) to prove or disprove the formulas in each case.

$$(\exists x.\ \forall y.\ R(x,y)) \to (\forall y.\ \exists x.\ R(x,y))$$
$$(\forall y.\ \exists x.\ R(x,y)) \to (\exists x.\ \forall y.\ R(x,y))$$
$$\exists x.\ \forall y,z.\ (P(y) \to Q(z)) \to (P(x) \to Q(x))$$
$$\neg(\exists y.\ \forall x.\ R(x,y) \leftrightarrow \neg(\exists z.\ R(x,z) \wedge R(z,x)))$$

> a) Negate and convert to clauses
>
> $$\begin{aligned} & \neg((\exists x.\ \forall y.\ R(x,y)) \to (\forall y.\ \exists x.\ R(x,y))) \\ \simeq\ & (\exists x.\ \forall y.\ R(x,y)) \wedge (\exists y.\ \forall x.\ \neg R(x,y)) && \text{(negate)} \\ \Longrightarrow\ & (\forall y.\ R(a,y)) \wedge (\forall x.\ \neg R(x,b)) && \text{(Skolemise)} \\ \Longrightarrow\ & R(a,y) \wedge \neg R(x,b) && \text{(drop } \forall\text{s)} \\ \Longrightarrow\ & \{R(a,y)\} \quad \{\neg R(x,b)\} \end{aligned}$$
>
> The two clauses can be unified with $[a/x, b/y]$, and resolved to the empty clause, proving the original formula.
>
> b) Negate and convert to clauses
>
> $$\begin{aligned} & \neg((\forall y.\ \exists x.\ R(x,y)) \to (\exists x.\ \forall y.\ R(x,y))) \\ \simeq\ & (\forall y.\ \exists x.\ R(x,y)) \wedge (\forall x.\ \exists y.\ \neg R(x,y)) && \text{(negate)} \\ \Longrightarrow\ & (\forall y.\ R(f(y),y)) \wedge (\forall x.\ \neg R(x,g(x))) && \text{(Skolemise)} \\ \Longrightarrow\ & R(f(y),y) \wedge \neg R(x,g(x)) && \text{(drop } \forall\text{s)} \end{aligned}$$

$$\implies \{R(f(y), y)\} \quad \{\neg R(x, g(x))\} \qquad \text{(convert to clauses)}$$

Unifying the first argument gives $[f(y)/x]$, but attempting to unify $g(x)[f(y)/x] = g(f(y))$ and y is not possible due to the occurs check. Since no resolution steps are possible, the original formula must be invalid.

c) Negate and convert to clauses

$$\begin{aligned}
&\neg(\exists x.\ \forall y, z.\ (P(y) \to Q(z)) \to (P(x) \to Q(x))) \\
&\simeq \forall x.\ \exists y, z.\ (P(y) \to Q(z)) \land P(x) \land \neg Q(x) && \text{(negate)} \\
&\simeq (\exists y, z.\ \neg P(y) \lor Q(z)) \land (\forall x.\ P(x) \land \neg Q(x)) && \text{(convert to miniscope)} \\
&\implies (\neg P(a) \lor Q(b)) \land (\forall x.\ P(x) \land \neg Q(x)) && \text{(Skolemise)} \\
&\implies (\neg P(a) \lor Q(b)) \land P(x) \land \neg Q(x) && \text{(drop } \forall\text{s)} \\
&\implies ① \{\neg P(a), Q(b)\} \quad ② \{P(x)\} \quad \neg ③ \{Q(x)\} && \text{(convert to clauses)}
\end{aligned}$$

Resolve ① and ② on $P(a)$ with the unifier $[a/x]$ to get ④ $\{Q(b)\}$. Resolve ④ and ③ on $Q(b)$ with the unifier $[b/x]$ to get $\square$.

d) Negate and convert to clauses

$$\begin{aligned}
&\neg\neg(\exists y.\ \forall x.\ R(x, y) \leftrightarrow \neg(\exists z.\ R(x, z) \land R(z, x))) \\
&\simeq \exists y.\ \forall x.\ R(x, y) \leftrightarrow \neg(\exists z.\ R(x, z) \land R(z, x)) && \text{(negate)} \\
&\simeq \exists y.\ \forall x.\ (R(x, y) \to \neg(\exists z.\ R(x, z) \land R(z, x))) \land ((\exists z.\ R(x, z) \land R(z, x)) \lor R(x, y)) \\
& && \text{(expand } \leftrightarrow\text{)} \\
&\simeq \exists y.\ \forall x.\ (\neg R(x, y) \lor (\forall z.\ \neg R(x, z) \lor \neg R(z, x))) \land ((\exists z.\ R(x, z) \land R(z, x)) \lor R(x, y)) \\
& && \text{(de Morgan)} \\
&\implies \forall x.\ (\neg R(x, a) \lor (\forall z.\ \neg R(x, z) \lor \neg R(z, x))) \land ((R(x, f(x)) \land R(f(x), x)) \lor R(x, a)) \\
& && \text{(Skolemise)} \\
&\implies (\neg R(x, a) \lor \neg R(x, z) \lor \neg R(z, x)) \land ((R(x, f(x)) \land R(f(x), x)) \lor R(x, a)) \\
& && \text{(drop } \forall\text{s)} \\
&\simeq (\neg R(x, a) \lor \neg R(x, z) \lor \neg R(z, x)) \land (R(x, f(x)) \lor R(x, a)) \land (R(f(x), x) \lor R(x, a)) \\
& && \text{(convert to CNF)} \\
&\implies ① \{\neg R(x, a), \neg R(x, z), \neg R(z, x)\} \ ② \{R(x, f(x)), R(x, a)\} \ ③ \{R(f(x), x), R(x, a)\} \\
& && \text{(convert to clauses)}
\end{aligned}$$

- There are no unit clauses, so we should see if factoring is possible. Indeed it is: the first clause has ④ $\{\neg R(a, a)\}$ as a factored instance.
- Resolve ④ and ② on $R(a, a)$ with the unifier $[a/x]$ to get ⑤ $\{R(a, f(a))\}$
- Resolve ④ and ③ on $R(a, a)$ with the unifier $[a/x]$ to get ⑥ $\{R(f(a), a)\}$
- Resolve ⑥ and ① on $R(f(a), a)$ and $R(x, a)$ with the unifier $[f(a)/x]$ to get ⑦ $\{\neg R(f(a), y), \neg R(y, f(a))\}$
- Resolve ⑥ and ⑦ on $R(f(a), a)$ and $R(f(a), y)$ with the unifier $[a/y]$ to get

⑧ $\{\neg R(a, f(a))\}$

- Resolve ⑧ and ⑤ on $R(a, f(a))$ to get $\square$.

5. Refute the following set of clauses using resolution and factoring.

$$① \{P(x,b), P(a,y)\} \qquad ② \{\neg P(x,b), \neg P(c,y)\} \qquad ③ \{\neg P(x,d), \neg P(a,y)\}$$

This is an example of when "greedy" factoring is problematic. We can factor all three clauses to get $\{P(a,b)\}$, $\{\neg P(c,b)\}$ and $\{\neg P(a,d)\}$, but there is no way to move forward with only these terms, since they have no variables left to unify. We need to be more strategic and only factor when we need to. Factoring ② yields $\{\neg P(c,b)\}$ which can be resolved with ① to get ④ $\{P(a,y)\}$. This, together with the factored clause ③, $\{\neg P(a,d)\}$, yields the empty clause, as required.

6. Prove the following formulas by resolution, showing all steps of the conversion into clauses. Note that P is just a predicate symbol, so in particular, x is not free in P.

$$(\forall x.\, P \vee Q(x)) \to (P \vee \forall x.\, Q(x)) \qquad \exists x, y.\, (R(x,y) \to \forall z, w.\, R(z,w))$$

a) Negating $(\forall x.\, P \vee Q(x)) \to (P \vee \forall x.\, Q(x))$ leaves the hypothesis untouched, so we can immediately read off the clause $\{P, Q(x)\}$. Negating the conclusion gives $\neg P \wedge \exists x.\, \neg Q(x)$, and Skolemising results in $\neg P \wedge \neg Q(a)$. The final clause set is:

$$① \{P, Q(x)\} \quad ② \{\neg P\} \quad ③ \{\neg Q(a)\}$$

Resolve ① and ③ on $Q(a)$ with the unifier $[a/x]$ to get ④ $\{P\}$, which, with ②, gives the empty clause.

b) Negating $\exists x, y.\, (R(x,y) \to \forall z, w.\, R(z,w))$ gives $\forall x, y.\, R(x,y) \wedge \exists z, w.\, \neg R(z,w)$. Skolemisation introduces two 2-place Skolem functions: $\forall x, y.\, R(x,y) \wedge \neg R(f(x,y), g(x,y))$. The two clauses are:

$$① \{R(x,y)\} \qquad ② \{\neg R(f(x,y), g(x,y))\}$$

While it seems like we cannot unify x with $f(x,y)$ due to the occurs check, we must always remember that variables in a clause can be renamed arbitrarily:

$$① \{R(u,v)\} \qquad ② \{\neg R(f(x,y), g(x,y))\}$$

Now, the two clauses resolve with the unifier $[f(x,y)/u, g(x,y)/v]$ and yield $\square$.

9. Optional exercises

1. In your own words, explain the motivation behind Herbrand interpretations.
   - How is a Herbrand interpretation constructed from a set of clauses S?
   - Why do we need Herbrand interpretations?
   - What is the significance of the Skolem–Gödel–Herbrand Theorem?

If you wish, consult a pre-2013 version of the course lecture notes, which discuss Herbrand models in more detail.

> This was explained in detail in Ex. 7.3 and the supplementary document.

2. Consider the Prolog program consisting of the definite clauses

$$P(f(x,y)) \leftarrow Q(x), R(y)$$
$$Q(g(z)) \leftarrow R(z)$$
$$R(a) \leftarrow$$

Describe the Prolog computation starting from the goal clause $\leftarrow P(v)$. Keep track of the substitutions affecting v to determine what answer the Prolog system would return.

> A Prolog program consists of a database of definite clauses (containing exactly one positive literal, representing the conclusion of an implication) and a goal clause (containing only negative literals, representing the set of unsolved goals). Computation happens by linear resolution: the goal clause is repeatedly resolved with one of the definite clauses until all goals are discharged.
>
> In this example, the Prolog program represents the following definite clauses:
>
> $$① \{\neg Q(x), \neg R(y), P(f(x,y))\} \quad ② \{\neg R(z), Q(g(z))\} \quad ③ \{R(a)\}$$
>
> The goal clause is Ⓖ $\{\neg P(v)\}$; the aim of the program is to find out what value for the variable v would give a contradiction. The advantages of the Prolog constraints are that one of the resolvents is always the result of the previous resolution (or the goal clause at the start), and the unifiers compose after each resolution step.
>
> - Resolve ① $\{\neg Q(x), \neg R(y), P(f(x,y))\}$ and Ⓖ with the unifier $[f(x,y)/v]$ to get Ⓖ $\{\neg Q(x), \neg R(y)\}$
> - Resolve ② $\{\neg R(z), Q(g(z))\}$ and Ⓖ with the unifier $[g(x)/x]$ to get Ⓖ $\{\neg R(z), \neg R(y)\}$
> - Resolve ③ $\{R(a)\}$ and Ⓖ with the unifier $[a/z]$ to get Ⓖ $\{\neg R(y)\}$
> - Resolve ③ $\{R(a)\}$ and Ⓖ with the unifier $[a/y]$ to get $\square$.
>
> The substitutions are:
>
> $$[f(x,y)/v] \circ [g(z)/x] \circ [a/z] \circ [a/y] = [f(g(a),a)/v, g(a)/x, a/y, a/z]$$
>
> That is, the final answer is $v = f(g(a),a)$.