Exercise 1: Negative strategy

In this exercise, we focus on propositional formulas. The signature is \( \mathcal{P} = \{ P_1(0), P_2(0), \ldots \} \) and \( \mathcal{F} = \emptyset \).

A clause is negative if it contains only negative literals. We study the following resolution strategy, called negative strategy: the application of the resolution rule is restricted to the case where one of the premisses is negative. We write \( \vdash \) the associated deduction relation.

1. Let \( E = \{ \neg P \vee Q, P \vee Q, P \vee \neg Q, \neg P \vee \neg Q \} \). Show that \( E \vdash \neg \bot \).

\[
\begin{array}{c}
\neg P \vee \neg Q & \quad P \vee \neg Q \\
\hline
\neg Q \vee \neg Q & \quad P \vee Q \\
\hline
\neg Q & \quad \neg P \\
\hline
P & \quad \bot \\
\end{array}
\]

2. Show that \( \geq_{lex} \) is an order and that for all partial interpretations \( I \) et \( J \), either \( I \leq J \), \( J \leq I \), \( I \leq_{lex} J \), or \( J \leq_{lex} I \).

Let us write \( I = (L_i)_{0 \leq i < n_i} \) and \( J = (N_j)_{0 \leq j < n_j} \). We have two possibilities:

- If \( n_i \geq n_j \), we have again two possibilities:
  - either \( J \subseteq I \), and so \( J \leq I \),
  - or \( I \subseteq J \), and so \( I \leq_{lex} J \).
– either there is a minimal \( k < n_j \) such that \( P_k \) is not in both \( I \) and \( J \). Then, depending on which interpretation contains \( P_k \), \( I \leq_{\text{lex}} J \), or \( J \leq_{\text{lex}} I \).

• The case where \( n_i < n_j \) is symmetric.

3. Let \( A \) be the semantic tree of a set of clauses \( E \). Assuming that \( A \) is finite and nonempty, show there exists a unique maximal partial interpretation for \( \leq_{\text{lex}} \) being a leaf and not falsifying any negative clause of \( E \).

First, there exists at least a partial interpretation not falsifying negative clauses in \( E \). As \( A \) is nonempty, it has at least one internal node \( I = (L_i)_{0 \leq i < n} \), which does not falsify any clause in \( E \). As a consequence, its leaf child \( L_\ast = (L_i)_{0 \leq i < n} \cup \{ \overline{A_n} \} \) does not falsify any negative clauses, so in particular not clause \( C \).

Finally, let \( P = \{ I_k : k \in K \} \) be the set of such leaves. As they are distinct leaves, for all pair \( k, k' \) such that \( k \neq k' \), \( I_k \not\leq_{\text{lex}} I_{k'} \) and \( I_{k'} \not\leq_{\text{lex}} I_{k} \).

By the previous question, either \( I_k \leq_{\text{lex}} I_{k'} \), either \( I_{k'} \leq_{\text{lex}} I_{k} \). The set \( P \) is finite and totally ordered by \( \leq_{\text{lex}} \), so it has a unique maximal element.

4. Prove the refutational completeness of \( \vdash \) using semantic trees.

Hint: consider the maximal leaf for \( \leq_{\text{lex}} \) not falsifying any negative clauses of \( E^\ast = \{ C : E \vdash \neg C \} \).

Another hint: \( I \) does not satisfy some formula of the form \( P_i \lor C \). Consider the tree rooted in partial interpretation \( I \cap \{ P_j, \overline{P_j} : 0 \leq j < i \} \cup \{ P_i \} \).

Let \( E^\ast \) be the set of clauses \( \{ C : E \vdash \neg C \} \). The set \( E^\ast \) is unsatisfiable as \( E \) is unsatisfiable. The semantic tree of \( E^\ast \) is as a consequence finite. Let us reason by contradiction and assume that \( A \) is nonempty, that is that \( \bot \) is not derivable from \( E \) with the negative strategy.

By the previous question, there is a unique maximal leaf \( I \) that does not falsify any negative clause of \( E^\ast \). Let \( C \) be the clause of \( E^\ast \) falsified by \( I \) with a minimal number of positive literals. We write \( C = P_i \lor C'' \). By the factorization rule and minimality of \( C \), \( P_i \) is the only occurrence of literal \( P_i \) in \( C \).

Let \( J \) be the partial interpretation \( I \cap \{ P_j, \overline{P_j} : 0 \leq j < i \} \). As \( I \) falsifies \( C \), \( P_i \not\in I \). Let us study the tree rooted in \( J \cup \{ P_i \} \), more specifically let us prove that it is a leaf. By contradiction, let us assume that it is not a leaf. Then it contains an internal node of maximal depth, with two children leaves. By the same reasoning as in the previous question, one of these leaves \( L' \) does not falsify any negative clause of \( E^\ast \). This is a contradiction, as \( L' >_{\text{lex}} L \).

As a consequence, \( J \cup \{ P_i \} \) is a leaf, and falsifies a negative clause of \( E^\ast \) of the form \( \neg P_i \lor C'' \). By the factorization rule, we can assume that \( C'' \) does not contain literals of the form \( \neg P_i \).

By the proof
clause $C' \lor C''$ is in set $E^*$. However, $J$ falsifies $C''$, so $I$ falsifies $C'$ and $C''$, so $I$ falsifies $C' \lor C'' \in E^*$ which contains one less positive literal than clause $P_i \lor C'$. This is a contradiction, as $J$ is not a leaf.
Exercise 2: Input strategy

We go back to first order, over a signature $\mathcal{P}, \mathcal{F}$.

We consider another resolution strategy: the resolution rule is restricted to the case where one of the premisses at least is in the set of initial clauses. This strategy is called the input strategy.

1. Show that $\bot$ cannot be derived from the set $E = \{ P \lor Q, \neg P \lor Q, P \lor \neg Q, \neg P \lor \neg Q \}$ using the input strategy. What is the consequence?

   The only clauses directly derivable from $E$ are $Q \lor Q$, $P \lor P$, $\neg P \lor \neg P$, and $\neg Q \lor \neg Q$. From these clauses and input resolution we can only derive clauses of $E$. Using factorization, we could generate $P$, $Q$, $\neg P$, and $\neg Q$. But again using input resolution, we cannot derive new clauses.

   As a consequence, the input strategy is not complete in general, even in the propositional case.

2. We say that a clause is a Horn clause if it contains at most one positive literal. Show that Horn clauses are stable by resolution and factorization.

   Let $C \lor P$ and $\neg P' \lor C'$ be two Horn clauses to which we can apply a resolution rule:
   
   \[
   \frac{C \lor P \quad \neg P' \lor C'}{\sigma(C \lor C') \text{ res}, \sigma = \text{mgu}(P, P')}
   \]

   As $P$ is a positive literal, $C$ is a negative clause. $C'$ is a Horn clause, so $\sigma(C \lor C')$ is a Horn clause.

   If $L \lor L' \lor C$ is a Horn clause, for any $\sigma$, $\sigma(L \lor C)$ is also a Horn clause.

Let $P, P', Q, Q'$ be literals, and $\sigma$ the most general unifier of unification problem $\{ P \overset{?}{=} P', Q \overset{?}{=} Q' \}$, i.e. a substitution $\sigma$ such that $\sigma(P) = \sigma(P')$ and $\sigma(Q) = \sigma(Q')$, and for any other such substitution $\theta$ there exists $\eta$ such that $\theta = \eta \circ \sigma$.

3. Let us assume that $\sigma_P = \text{mgu}(P, P')$ and $\sigma_Q = \text{mgu}(\sigma_P(Q), \sigma_P(Q'))$. Prove that $\sigma = \sigma_Q \circ \sigma_P$.

   First, let us prove that $\sigma' = \sigma_Q \circ \sigma_P$ is a unifier of $\{ P \overset{?}{=} P', Q \overset{?}{=} Q' \}$:

   - as $\sigma_P(P) = \sigma_P(P')$, we directly have $\sigma'(P) = \sigma'(P')$,
   - By definition of $\sigma_Q$, $\sigma'(Q) = \sigma'(Q')$.

   Let $\theta$ be another unifier. There is $\eta$ such that $\theta = \eta \circ \sigma_P$, as $\theta$ unifies $P$ and $P'$. We also have $\theta(Q) = \theta(Q')$, so $(\eta \circ \sigma_P)(Q) = (\eta \circ \sigma_P)(Q')$. Substitution $\eta$ unifies $\sigma_P(Q)$ and $\sigma_P(Q')$, so there is $\eta'$ such that $\eta = \eta' \circ \sigma_Q$. As a consequence, $\theta = \eta' \circ \sigma'$. 
The goal of the exercise is to show that the input strategy is refutationally complete for Horn clauses. We fix a set of Horn clauses $E$ and we want to show by induction that for every resolution proof $\pi$ of a clause $C$ from $E$, there is a proof $\pi'$ of $C$ following the input strategy from $E$. To this end, we define:

- $N(\pi)$, the number of nodes in $\pi$,
- $H(\pi)$, the number of nodes in the left-hand subproof of $\pi$ (if $\pi$ starts with a factorization, the number of nodes in its only subproof).

4. Prove that the input strategy on Horn clauses is refutationally complete. Proceed by induction on $(N(\pi), H(\pi))$, ordered lexicographically, to transform any derivation from Horn clauses using resolution into a derivation using the input strategy.

Hint: The problematic case is the one where the first rule $R$ in $\pi$ is a resolution. In this case, make a disjunction of cases on the first rule $R'$ of the left-hand subproof, and permute the two rules $R, R'$ to diminish the size of the left-hand subproof.

We proceed by induction on proof $\pi$ of clause $C$ from $E$.

If $N(\pi) = 0$, then the proof is a leaf.

Else if $H(\pi) = 0$, then one of the premisses of the last applied rule is a leaf. We apply the induction hypothesis to the right-hand subproof (if it exists), so we have a proof using the input strategy of $C$.

Else if the last rule of $\pi$ is a factorization, we apply the induction hypothesis to its subproof.

Else, the last rule is a resolution, and we study the last rule of its left-hand side proof $\pi'$:

- If the last rule in $\pi'$ is a resolution rule, $\pi$ is of the form:

$$
\begin{align*}
C_1 \lor P & \quad \neg P' \lor C_2 \lor Q \\
\sigma_P(C_1 \lor C_2 \lor Q) & \quad \neg Q' \lor C_3 \\
\sigma_Q(\sigma_P(C_1 \lor C_2) \lor C_3) & \quad \text{res}
\end{align*}
$$

As we can assume that $fv(P) \cap fv(Q', Q, P') = \emptyset$, we have that $\sigma_P(Q') = Q'$ we can apply the previous question. Defining $\sigma = \sigma_Q \circ \sigma_P$, it is the mgu of unification problem $\{ P \not= P', Q \not= Q' \}$. As a consequence, $Q$ and $Q'$ are unifiable, let us write $\sigma_{Q'} = mgu(Q, Q')$. We can apply the rule:

$$
\begin{align*}
\neg P' \lor C_2 \lor Q & \quad \neg Q' \lor C_3 \\
\sigma_{Q'}(P' \lor C_2 \lor C_3) & \quad \text{res}
\end{align*}
$$

Now to apply last question again we just have to prove that $\sigma_{Q'}(P')$ and $\sigma_Q(P)$ are unifiable. As $\sigma$ unifies $Q, Q'$, there is
η such that σ = η ◦ σ_Q'. As \( f v(P) \cap f v(Q', Q, P') = \emptyset \) and σ unifies \( P, P' \), η unifies \( σ_Q(P') \) and \( P \). We can apply the previous question and build the proof:

\[
\begin{align*}
\begin{array}{c}
\neg P' \lor C_2 \lor Q \\
C_1 \lor P \\
\sigma_Q'(P' \lor C_2 \lor C_3)
\end{array}
\end{align*}
\]

res

\[
\begin{align*}
\sigma(C_1 \lor C_2 \lor C_3)
\end{align*}
\]

The number of nodes in the left-hand side of this proof is strictly smaller than that of the initial proof. We can apply the induction hypothesis.

- If the last rule in \( π' \) is a factorization rule, \( π \) is of the form:

\[
\begin{align*}
\neg P' \lor C_2 \lor Q \\
\sigma_P(\neg P \lor C_1 \lor Q) \\
\sigma_Q(\sigma_P(\neg P \lor C_1) \lor C_2)
\end{align*}
\]

As we can assume that \( f v(P) \cap f v(\neg Q' \lor C_2) = \emptyset \), we can apply the previous question. Defining \( σ = σ_Q \circ σ_P \), it is the mgu of unification problem \( \{ P \overset{?}{=} P', Q \overset{?}{=} Q' \} \). As a consequence, \( Q \) and \( Q' \) are unifiable, let us write \( σ_Q = mgu(Q, Q') \). We can apply the rule:

\[
\begin{align*}
\neg P \lor \neg P' \lor C_1 \lor Q \\
\sigma_Q(\neg P \lor \neg P' \lor C_1 \lor C_2)
\end{align*}
\]

res

Now to apply last question again we just have to prove that \( σ_Q'(P') \) and \( σ_Q'(P) \) are unifiable. As σ unifies \( Q, Q' \), there is η such that \( σ = η \circ σ_Q \). As σ unifies \( P, P' \), η unifies \( σ_Q'(P') \) and \( σ_Q'P \). We can apply the previous question and build the proof:

\[
\begin{align*}
\neg P \lor \neg P' \lor C_1 \lor Q \\
\sigma_Q(\neg P \lor \neg P' \lor C_1 \lor C_2)
\end{align*}
\]

res

\[
\begin{align*}
\sigma(\neg P \lor C_1 \lor C_2)
\end{align*}
\]

fact

By induction, the input strategy is refutationally complete.
Exercise 3: Security
We want to represent cryptographic protocols using Horn clauses. We will proceed using the following signature:

- Terms represent messages exchanged by participants.
- Cryptographic primitives are represented by functions:
  - pair(2) and aenc(2) are binary function symbols representing respectively pairs of messages and encryption of a message using a key.
  - pk(1) is a unary function symbol representing the public key of a participant.
  - s(0) is a constant function symbol representing a secret.
  - a(0), b(0), i(0) are constant function symbols representing the secret keys of the three participants Alice, Bob, and Impostor (the attacker).
- The attacker and her abilities are represented by a unary predicate att(1).

For example, the attacker can construct and deconstruct pairs, represented by Horn clauses in the following way:

\[
\text{att}(x) \land \text{att}(y) \Rightarrow \text{att}(\text{pair}(x, y)) \\
\text{att}(\text{pair}(x, y)) \Rightarrow \text{att}(x) \\
\text{att}(\text{pair}(x, y)) \Rightarrow \text{att}(y)
\]

She can also, given a public key, encrypt messages:

\[
\text{att}(m) \land \text{att}(k) \Rightarrow \text{att}(\text{aenc}(m, k))
\]

To decrypt messages, she needs the secret key of the associated participant:

\[
\text{att}(\text{aenc}(m, \text{pk}(p))) \land \text{att}(p) \Rightarrow \text{att}(m)
\]

We also assume that the attacker has access to the public keys of other participants, represented by clauses att(pk(a)) and att(pk(b)), and that she has her own secret and public keys, represented by clauses att(i) and att(pk(i)). We name \( A \) the set of 9 clauses we just described.

1. Prove that if the attacker has access to an encrypted secret but not to the associated secret key, she cannot get the secret, i.e. one cannot derive \( \bot \) from \( A \cup \{ \text{att(aenc}(s, \text{pk}(a))) , \neg \text{att}(s) \} \) using resolution.

   Hint: use resolution by selection, with selection function choosing literals of the form att(t) or \( \neg \text{att}(t) \) where \( t \) is not a variable when possible, else a positive literal, else an arbitrary literal.

The only clauses derivable with this selection strategy are \( \neg \text{att}(a) \lor \text{att}(s) \) and \( \neg \text{att}(a) \). Clause \( \bot \) is not derivable.

Here is a cryptographic protocol:
• Participant $A$ contacts participant $B$, encrypting with $B$’s public key both her public key and a secret already encrypted with $B$’s public key:

$$A \rightarrow B : \text{pair}(pk(a), \text{aenc}(s, pk(b)))$$

• Participant $B$ responds with the secret encoded with $A$’s public key:

$$B \rightarrow A : \text{aenc}(s, pk(a))$$

The attacker can intercept messages exchanged during this protocol, and transmit messages to Alice and Bob, which is represented by the Horn clauses:

$$\text{att}(\text{pair}(pk(a), \text{aenc}(s, pk(b))))$$
$$\text{att}(\text{pair}(x, \text{aenc}(y, pk(b)))) \Rightarrow \text{att}(\text{aenc}(y, x))$$

We call $P$ the set of 11 clauses containing $A$ and the two above clauses.

2. Prove that an attack is possible on this protocol, i.e. one can derive $\bot$ from $P \cup \{\neg \text{att}(s)\}$.

Intuitively, the attacker will intercept the message of $A$:  

$$\text{att}(\text{pair}(pk(a), \text{aenc}(s, pk(b)))) \quad \neg \text{att}(\text{pair}(x, y)) \lor \text{att}(y)$$

Then reconstructs the message with her own key:

$$\text{att}(\text{aenc}(s, pk(b))) \quad \text{att}(pk(i))$$

Then pretends to be $A$ and sends the message to $B$:

$$\text{att}(\text{pair}(pk(i), \text{aenc}(s, pk(b)))) \quad \neg \text{att}(\text{pair}(x, \text{aenc}(y, pk(b)))) \lor \text{att}(\text{aenc}(y, x))$$

Finally, she decrypts the secret with her own key:

$$\text{att}(\text{aenc}(s, pk(i))) \quad \neg \text{att}(\text{aenc}(x, pk(y))) \lor \neg \text{att}(y) \lor \text{att}(x)$$

3. A way to prevent this attack is to encrypt pair of the identity of $A$ and the secret in the first message, yielding protocol

$$A \rightarrow B : \text{aenc}(\text{pair}(pk(a), s), pk(b))$$
$$B \rightarrow A : \text{aenc}(s, pk(a))$$

and associated clauses

$$\text{att}(\text{aenc}(\text{pair}(pk(a), s), pk(b)))$$
$$\text{att}(\text{aenc}(\text{pair}(x, y), pk(b))) \Rightarrow \text{att}(\text{aenc}(y, x))$$
Let \( P' \) be the set of eleven clauses containing \( A \) and the two above clauses. Show that we cannot derive \( \bot \) from \( P' \cup \{ \neg \text{att}(s) \} \).

*Hint: use resolution by selection with the same selection function as in the question 1 hint.*

The only clauses derivable with this selection strategy are:

\[
\begin{align*}
\neg \text{att}(b) \lor \text{att}(\text{pair}(\text{pk}(a), s)) \\
\neg \text{att}(b) \lor \text{att}(\text{pk}(a)) \\
\neg \text{att}(b) \lor \text{att}(s) \\
\neg \text{att}(b) \\
\text{att}(\text{aenc}(s, \text{pk}(a))) \\
\neg \text{att}(a) \lor \text{att}(s) \\
\neg \text{att}(a)
\end{align*}
\]

Clause \( \bot \) is not derivable.