Logic in CS Autumn 2024

Problem Sheet 3

S. Krishna

1. For each of the following conditions, give an example of an unsatisfiable set of formulae, Γ that meets the condition.

   (a) Each member of the set is—by itself—satisfiable.

   (b) For any two members γ_1 and γ_2 of Γ, the set $\{\gamma_1, \gamma_2\}$ is satisfiable.

   (c) For any three members γ_1, γ_2, and γ_3 of Γ, the set $\{\gamma_1, \gamma_2, \gamma_3\}$ is satisfiable.

Solution

(a) $\Gamma = \{p, \neg p\}$

(b) $\Gamma = \{p_1, p_2, \neg p_1 \vee \neg p_2\}$

(c) $\Gamma = \{p_1, p_2, p_3, \neg p_1 \vee \neg p_2 \vee \neg p_3\}$

2. Let α be a wff whose only connective symbols are $\wedge$, $\vee$, and $\neg$. Let α^* be the result of interchanging $\wedge$ and $\vee$ and replacing each propositional variable by its negation. Show that α^* is tautologically equivalent to $\neg\alpha$. Observe that this result is a generalization and a stronger form of De Morgan's laws, which deal with the negation of conjunctions and disjunctions.

Solution

We will prove this via structural induction.
Base Case:
Consider the simplest wff, where α is a sentence symbol p. Then, $\alpha^* = \neg p$, and hence, $\alpha^* \equiv \neg\alpha$.

Inductive Step:
Say ϕ, ψ are formulae which such that $\phi* \equiv \neg\phi$ and $\psi* \equiv \neg\psi$. Consider the following cases:

- $\alpha = \neg\phi$: Then, $\alpha^* = \neg\phi^*$ and by the inductive hypothesis, $\phi^* \equiv \neg\phi$.Therefore, $\alpha^* = \neg(\neg\phi) = \phi$, and thus $\alpha^* \equiv \neg\alpha$.
- $\alpha = \phi \wedge \psi$: Then, by interchanging $\vee$ and $\wedge$, $\alpha^* = \phi^* \vee \psi^*$, and by the inductive hypothesis, $\phi^* \equiv \neg\phi$ and $\psi^* \equiv \neg\psi$. Thus, $\alpha^* = \neg\phi \vee \neg\psi$ and by De Morgan's law, $\alpha^* \equiv \neg\alpha$.
- $\alpha = \phi \vee \psi$: Then, by interchanging $\vee$ and $\wedge$, $\alpha^* = \phi^* \wedge \psi^*$, and by the inductive hypothesis, $\phi^* \equiv \neg\phi$ and $\psi^* \equiv \neg\psi$. Thus, $\alpha^* = \neg\phi \wedge \neg\psi$ and by De Morgan's

law, $\alpha^* \equiv \neg\alpha$.

3. Let $\mathcal{F}$ and $\mathcal{G}$ be two sets of formulae. We say $\mathcal{F} \equiv \mathcal{G}$ iff for any assignment α, $\alpha \models \mathcal{F}$ iff $\alpha \models \mathcal{G}$ ($\alpha \models \mathcal{F}$ iff $\alpha \models F_i$ for every $F_i \in \mathcal{F}$). Prove or disprove: For any $\mathcal{F}$ and $\mathcal{G}$, $\mathcal{F} \equiv \mathcal{G}$ iff

   (a) For each $G \in \mathcal{G}$, there exists $F \in \mathcal{F}$ such that $G \models F$, and

   (b) For each $F \in \mathcal{F}$, there exists $G \in \mathcal{G}$ such that $F \models G$,

Solution

Consider $\mathcal{F} = \{p\}$ and $\mathcal{G} = \{p, \top\}$, where p is an atomic proposition. Observe that $\mathcal{F} \equiv \mathcal{G}$. *(Why?)* Now, we shall show that (a) is false, hence giving a counterexample.

Take $G = \top$. Clearly, $\top \not\models p$ and hence there does not exist a $F \in \mathcal{F}$ such that $G \models F$

4. A set of sentences $\mathcal{F}$ is said to be closed under conjunction if for any F and G in $\mathcal{F}$, $F \wedge G$ is also in $\mathcal{F}$. Suppose $\mathcal{F}$ is closed under conjunction and is inconsistent ($\mathcal{F} \vdash \bot$). Prove that for any $G \in \mathcal{F}$, there exists $F \in \mathcal{F}$ such that $\{F\} \vdash \neg G$.

Solution

We are given that $\mathcal{F}$ is inconsistent ($\mathcal{F} \vdash \bot$). Let $\mathcal{F}' \subseteq \mathcal{F}$ be the finite set of formulae in $\mathcal{F}$ used in a proof deducing $\bot$. Thus, $\mathcal{F}'$ is a finite subset of $\mathcal{F}$ such that $\mathcal{F}' \vdash \bot$.

Consider the formula $F = \bigwedge_{F' \in \mathcal{F}'} F'$. Since $\mathcal{F}'$ is a finite subset of $\mathcal{F}$, F is also an element of $\mathcal{F}$ since $\mathcal{F}$ is closed under conjunction. We shall show that $\{F\} \vdash \bot$.

$$
\begin{aligned}
\mathcal{F}' \vdash \bot &\implies \mathcal{F}' \models \bot \qquad [\text{ Soundness of Formal Proof System }] \\
&\implies \nexists\, \alpha \text{ such that } (\forall F' \in \mathcal{F}',\ \alpha \models F') \\
&\implies \nexists\, \alpha \text{ such that } \alpha \models F \qquad [\because F = \bigwedge_{F' \in \mathcal{F}'} F' \,] \\
&\implies \{F\} \models \bot \implies \{F\} \vdash \bot \quad [\text{ Completeness of Formal Proof System }]
\end{aligned}
$$

Hence, we showed that there exists a formula $F \in \mathcal{F}$ such that $\{F\} \vdash \bot$ and hence for any formula $G \in \mathcal{F}$, $\{F\} \vdash \neg G$ since $\bot \vdash \neg G$.

5. Suppose $\models (F \to G)$ and F is satisfiable and G is not valid. Show that there exists a formula H such that the atomic propositions in H are in both F and G and $\models F \to H$ and $\models H \to G$.

Solution

We denote the list of propositional variables that occur only in F as $\vec{p}$, the list of variables common to F and G as $\vec{q}$, and those only in G as $\vec{r}$. Let n_p, n_q, n_r be the number of variables in $\vec{p}, \vec{q}, \vec{r}$, respectively.
Define H as follows:

$$H = \bigvee_{\vec{\alpha} \in \{\bot, \top\}^{n_p}} F[\, \vec{p} = \vec{\alpha}\,].$$

where, $F[\, \vec{p} = \vec{\alpha}\,]$ denotes the formula obtained by replacing all occurrences of p_i by α_i, and then simplifying the resultant formula. By simplification, we mean the obvious ones like $\alpha \wedge \top = \alpha$, $\alpha \vee \top = \top$, $\alpha \wedge \neg\top = \bot$, $\alpha \vee \neg\top = \alpha$ for all sub-formulas α of F. Similarly for $\bot$.

Notice that H contains variables only in $\vec{q}$.
Now we need to show that (a) $\models F \to H$ and (b) $\models H \to G$. In the following discussion, α_p (resp. α_q and α_r) denote an assignment of variables in $\vec{p}$ (resp. $\vec{q}$ and $\vec{r}$). For this we will state the following lemma first.

Lemma. $(-, \alpha_q, -) \models H \iff \exists\, \alpha_p$ such that $(\alpha_p, \alpha_q, -) \models F$.
Here, $(-, \alpha_q, -)$ represents an assignment of all variables such that the variables in $\vec{q}$ are assigned α_q, while variables in $\vec{p}$ and $\vec{r}$ can take any truth values. Thus, the lemma follows from the definition of H and the fact that $[\![H]\!]$ does not depend on assignment of variables in $\vec{p}$.

(a) $\models F \to H$:

$$\begin{aligned}(\alpha_p, \alpha_q, -) \models F &\implies \exists\, \alpha_p \text{ such that } (\alpha_p, \alpha_q, -) \models F \\ &\implies (-, \alpha_q, -) \models H \quad [\text{ By } (\impliedby) \text{ of Lemma }]\end{aligned}$$

(b) $\models H \to G$:

$$\begin{aligned}(-, \alpha_q, -) \models H &\implies \exists\, \alpha_p \text{ such that } (\alpha_p, \alpha_q, -) \models F \quad [\text{ By } (\implies) \text{ of Lemma }] \\ &\implies \exists\, \alpha_p \left(\forall\, \alpha_r,\ (\alpha_p, \alpha_q, \alpha_r) \models F\right) \quad [\, [\![F]\!] \text{ does not depend on } \alpha_r\,] \\ &\implies \exists\, \alpha_p \left(\forall\, \alpha_r,\ (\alpha_p, \alpha_q, \alpha_r) \models G\right) \quad [\text{ Since } \models F \to G\,] \\ &\implies \forall \alpha_r,\ (-, \alpha_q, \alpha_r) \models G \quad [\, [\![G]\!] \text{ does not depend on } \alpha_p\,]\end{aligned}$$

This result is popularly known as *Craig's Interpolation Theorem*, as applied to propositional logic. The formula H is known as an *interpolant* of F and G.

6. Consider the parity function, $\mathsf{PARITY} : \{0,1\}^n \to \{0,1\}$, where PARITY evaluates to 1 iff an odd number of inputs is 1. In all of the CNFs below, we assume that each clause contains any variable at most once, i.e. no clause contains expressions of the form $p \wedge \neg p$ or $p \vee \neg p$. Furthermore, all clauses are assumed to be distinct.

(a) Prove that any CNF representation of PARITY must have n literals (from distinct variables) in every clause.

(b) Prove that any CNF representation of PARITY must have at least 2^{n-1} clauses.

Solution

Let $\mathsf{PARITY} := \bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} \ell_{ij}$ be the CNF representation of PARITY, where n_i is the number of literals in the $i-$th clause, and m is the number of clauses. We want to prove that $n_i = n$ for every i, and $m \geq 2^n - 1$.

(a) Suppose $n_i < n$ for some i. Negate the entire formula to convert the CNF into a DNF. Now, choose an assignment of literals in the $i-$th cube (of the DNF) such that the cube, and hence the whole DNF formula, evaluates to 1. Now, consider a variable v, which doesn't appear in the $i-$th cube, and flip its value. The LHS then becomes 0. However, the $i-$th clause stays 1, leading to a contradiction.

(b) Once again, consider the DNF. Note that a conjunction of n literals is satisfied by a unique assignment of variables, and thus, the clauses in the DNF actually encode the assignments that satisfy the formula. Since PARITY is satisfied by 2^{n-1} clauses, we're done.

7. Using resolution, or otherwise, show that there is a polynomial-time algorithm to decide satisfiability of those CNF formulas F in which each propositional variable occurs at most twice. Justify your answer. Note that this question is not the same as 2-SAT.

Solution

If a variable occurs only positively or only negatively in F, then we can delete the clauses containing that variable without affecting the satisfiability of F. Thus, we can assume that all variables have one negative occurrence and one positive occurrence.

Suppose F has clauses C_1, C_2 such that $p \in C_1$ and $\neg p \in C_2$. Let $R = (C_1 \backslash \{p\}) \cup (C_2 \backslash \{\neg p\})$ be a resolvent of C_1, C_2. We claim that F and $F' = (F \backslash \{C_1, C_2\}) \cup \{R\}$ are equisatisfiable. It follows immediately from resolution that a valuation that satisfies F also satisfies F'. Thus, it suffices to show that an assignment α that satisfies F' can be extended to an assignment that satisfies F. Since such an assignment satisfies R, it satisfies either $C_1 \backslash \{p\}$ or $C_2 \backslash \{\neg p\}$. In the first case, the assignment $\alpha[p \leftarrow 0]$ satisfies F, and in the second case, the assignment $\alpha[p \leftarrow 1]$ satisfies F.

In summary, we can eliminate each variable from F without affecting satisfiability and without increasing the size of the overall formula. It follows that satisfiability can be decided in polynomial time.

8. Say that a set Σ_1 of wffs is equivalent to a set Σ_2 of wffs iff for any wff α, we have $\Sigma_1 \models \alpha$ iff $\Sigma_2 \models \alpha$. A set Σ is independent iff no member of Σ is tautologically implied by the remaining members in Σ. Show that a finite set of wffs has an independent equivalent subset by describing an algorithm to compute this independent equivalent subset. Prove that your algorithm returns a subset that is independent and equivalent.

Algorithm 1 Algorithm to compute an independent equivalent subset

1: $\Sigma' \leftarrow \Sigma$
2: **while** there exists $\sigma \in \Sigma'$ such that $\Sigma' \setminus \{\sigma\} \models \sigma$ **do**
3: $\quad \Sigma' \leftarrow \Sigma' \setminus \{\sigma\}$
4: **end while**
5: **return** Σ'

Solution

The key idea is to keep removing the formulae in Σ, which are tautologically implied by others, till there aren't any. This is given in Algorithm 1.

Notice that since Σ is finite, the algorithm terminates after finite number of iterations. Also, the algorithms terminates only when there does not exist a $\sigma \in \Sigma'$ such that $\Sigma \setminus \{\sigma\} \models \sigma$, i.e, the set Σ' returned at the end will be independent. Clearly $\Sigma' \subseteq \Sigma$. We are only left to show that Σ' is equivalent to Σ.

Let the algorithm terminate after n iterations. Let $\Sigma_0, \Sigma_1, \ldots, \Sigma_n$ be the sets representing Σ' at the end of each iteration. Clearly, $\Sigma_0 = \Sigma$, and Σ_n will be returned at the end. We will inductively show that at each iteration, i, Σ_i is equivalent to Σ.

Base case. $i = 0$. $\Sigma_i = \Sigma_0$ is clearly equivalent to Σ.
Induction Step. Assume that Σ_{i-1} is equivalent to Σ. We will now show that Σ_i is also equivalent to Σ. In our algorithm, we should have removed an element from Σ_{i-1} to obtain Σ_i. Let this element be σ_i. We have removed σ_i from Σ_{i-1} because, $\Sigma_i \models \sigma_i$. Now, let α be any wff. If $\Sigma_i \models \alpha$, then since $\Sigma_i \subseteq \Sigma$, $\Sigma \models \alpha$ (*Why?*).
Now, observe that since, $\Sigma_{i-1} \models \sigma_i$, the set of all assignments which make all formulae in Σ_{i-1} `true` is same as that which make all formulae in $\Sigma_i = \Sigma_{i-1} \setminus \{\sigma_i\}$ `true`. Hence, since Σ is equivalent to Σ_{i-1} it follows that $\Sigma \models \alpha \implies \Sigma_{i-1} \models \alpha \implies \Sigma_i \models \alpha$.
Thus, Σ_i is equivalent to Σ. Thus by induction Σ_n is equivalent to Σ, as we needed, and thus our algorithm returns an independent equivalent subset of Σ.

Follow-up Question. Is the output from our algorithm unique?