Homework 2 Solutions

1. The Case of Dr. Equisemantic and Mr. Irredundant 15 points

Assume we have a countably infinite list of propositional variables $p_1, p_2, \ldots$. For this problem, by "formula", we always mean a finite string representing "syntactically-correct formula". Let Σ be a set of formulae. For any formula φ, we say $\Sigma \models \varphi$ (read as Σ semantically entails φ) if for any assignment α of the propositional variables that makes all the formulae contained in Σ true, α also makes φ true.

Let us call two sets of formulae Σ_1 and Σ_2 *equisemantic* if for every formula φ, we have $\Sigma_1 \models \varphi$ if and only if $\Sigma_2 \models \varphi$. Furthermore, let us call a non-empty set of formulae Σ *irredundant* if no formula σ in Σ is semantically entailed by $\Sigma \setminus \{\sigma\}$.

1. *[5 points]* Show that any set of formulae Σ must always be countable. This implies that we can enumerate the elements of Σ. Assume from now on that $\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \ldots\}$.
2. Suppose we define Σ' as follows:

$$\begin{aligned}\Sigma' = \{&\sigma_1,\\ &(\sigma_1) \to \sigma_2,\\ &(\sigma_1 \wedge \sigma_2) \to \sigma_3,\\ &(\sigma_1 \wedge \sigma_2 \wedge \sigma_3) \to \sigma_4,\\ &\vdots\\ \}&\end{aligned}$$

   Suppose we remove all the tautologies from Σ' and call this reduced set Σ''. Prove that Σ'' is irredundant and equisemantic to Σ. You can proceed as follows:

   (a) *[5 points]* Show that a non-empty satisfiable set Γ with $|\Gamma| \geq 2$ is irredundant if and only if $(\Gamma \setminus \{\gamma\}) \cup \{\neg\gamma\}$ is satisfiable for every $\gamma \in \Gamma$.

   (b) *[5 points]* Use the above result to show that Σ'' is irredundant and equisemantic to Σ.

Solution:

1. Recall from your course on Discrete Structures that a set is *countable* if either it is finite, or if there is a bijection between the set and the set of natural numbers. Equivalently, a set is countable if there is an injective function from the set to the set of natural numbers. Recall also that the Cartesian product of two countable sets is countable, and the countable union of countable sets is also countable.

   Now, consider the set S of all finite strings. This set must be countable. To see why this is so, take the set S_1 of all strings of length 1. This set is countable because the number of propositional variables is countable and we only have an additional finite set of non-variable symbols (parentheses, logical and symbol, etc.). Now consider the set S_2 of all strings of length 2. Being the Cartesian product of two countable sets ($S_1 \times S_1$), this set must also be countable. Inductively, the set S_n of all strings of length n, for any $n \geq 2$, is the Cartesian product of two countable sets ($S_{n-1} \times S_1$), and hence must be countable. Clearly $S = \cup_{n=1}^{\infty} S_n$. Being a countable union of countable sets, the set S must therefore be countable.

The set of all syntactically-correct formulas is a (strict) subset of the set of all possible finite strings. Being a subset of a countable set, it must be countable. Σ, in turn, is a subset of the set of all syntactically-correct formulas. Being a subset of a countable set, Σ must be countable.

2. (a) Note first that since Γ is satisfiable, so is $\Gamma \setminus \{\gamma\}$ (every α that satisfies all formulas in Γ certainly satisfies all formulas in $\Gamma \setminus \{\gamma\}$). Now, if γ is not semantically entailed by $\Gamma \setminus \{\gamma\}$, there must be some assignment α that makes each formula in $\Gamma \setminus \{\gamma\}$ true but makes γ false. This same assignment α would thus make $\neg\gamma$ true, and hence would be a satisfying assignment for $(\Gamma \setminus \{\gamma\}) \cup \{\neg\gamma\}$. If this holds for every $\gamma \in \Gamma$, this means that no element of Γ is semantically entailed by the rest of the elements. Hence Γ is irredundant.

   To prove the other direction, suppose Γ is irredundant. By definition, no formula $\gamma \in \Gamma$ is semantically entailed by $\Gamma \setminus \{\gamma\}$. In other words, for every $\gamma \in \Gamma$, there exists an assignment α (dependent on γ in general) that satisfies every formula in $\Gamma \setminus \{\gamma\}$, but does not satisfy γ. Hence, this α satisfies all formulas in $(\Gamma \setminus \{\gamma\}) \cup \{\neg\gamma\}$. Since the above argument holds for all $\gamma \in \Gamma$, this proves the statement.

   (b) The fact that Σ'' is equisemantic to Σ is easy to see. We will first show that an assignment α satisfies all formulas in Σ iff it satisfies all formulas in Σ". Clearly, if each of the formulas in Σ evaluates to true for α, then both sides of each implication in Σ' evaluate to true for α. Hence, each of the formulas in Σ' is also true for the same assignment. Conversely, if each of the formulas in Σ' is true for assignment α, then σ_1 is true, and since $(\sigma_1) \to \sigma_2$ is true, this implies σ_2 is true. Using the same reasoning, it follows by induction that σ_n is true for all $n \geq 1$. Thus, each formula in Σ is true for the assignment α.

   Let S denote the set of satisfying assignments of Σ. We have just shown above that S is also the set of satisfying assignments of Σ'. Now suppose $\Sigma \models \varphi$. This is equivalent to saying that every assignment in S satisfies φ. Since S is also the set of satisfying assignments of Σ', this is equivalent to saying that $\Sigma' \models \varphi$. Hence, if $\Sigma \models \varphi$, then $\Sigma' \models \varphi$ too. A similar reasoning shows the result the other way round. Hence, Σ and Σ' are equisemantic.

   Since Σ'' is just Σ' without the tautologies, and tautologies, being always true, do not change equisemanticness, Σ'' is equisemantic to Σ.
   Let us now look at irredundancy. Let $\Sigma'' = \{\eta_1, \eta_2, \ldots\}$ in order after removing the tautologies from Σ'. Consider any $\eta_n \in \Sigma''$. Since η_n is not a tautology, there must be an assignment α that makes η_n false. By the semantics of $\to$, this assignment must make each of $\sigma_1, \ldots, \sigma_{n-1}$ true and σ_n false. Since α makes each of $\sigma_1, \ldots \sigma_{n-1}$ true, it satisfies all implications $\eta_1, \ldots \eta_{n-1}$ (both sides of implication having formulas in $\{\sigma_1, \ldots \sigma_{n-1}\}$, must evaluate to true). Similarly, α satisfies all implications $\eta_{n+1}, \ldots$, since the left side of each of these implications has σ_n conjuncted with other formualas, and σ_n evaluates to false under assignment α. Therefore, α is a satisfying assignment for $\Sigma'' \setminus \{\eta_n\}$. Since α also falsifies η_n, it immediately follows that α satisfies all formulas in $(\Sigma'' \setminus \{\eta_n\}) \cup \{\neg\eta_n\}$. Since the abov argument holds for every $\eta_n \in \Sigma$", we conclude that Σ'' has no element that is semantically entailed by the others. Hence Σ" is irredundant.

2. A follow-up of the take-away question of Tutorial 2 — 25 points

To recap from the take-away question of Tutorial 2, we will view the set of assignments satisfying a set of propositional formulae as a language and examine some properties of such languages.

Let $\mathbf{P}$ denote a countably infinite set of propositional variables $p_0, p_1, p_2, \ldots$. Let us call these variables positional variables. Let Σ be a countable set of formulae over these positional variables. Every assignment $\alpha : \mathbf{P} \to \{0,1\}$ to the positional variable can be uniquely associated with an infinite bitstring w, where the i^{th} bit $w_i = \alpha(p_i)$. The language defined by Σ, also called $L(\Sigma)$, is the set of bitstrings w for which the corresponding assignment α, that has $\alpha(p_i) = w_i$ for each i, satisfies Σ, that is, for each formula $F \in \Sigma$, $\alpha \models F$. In this case, we say that $\alpha \models \Sigma$. Let us call the languages definable in this manner as PL-definable languages.

Example:

Let $\Sigma = \{p_0 \to p_1, p_1 \to p_2, p_2 \to p_3, \ldots\}$.

Then $L(\Sigma) = \{1111\ldots, 0111\ldots, 0011\ldots, 0001\ldots, \ldots, 0000\ldots\}$, or, to be precise, if we denote the infinite bitstring containing only 1s by 1^ω and the infinite bitstring containing only 0s by 0^ω, and the finite bitstring consisting of k 0s by 0^k, then $L(\Sigma) = \{0^k 1^\omega : k \in \mathbb{N}\} \cup \{0^\omega\}$.

(a) *[5 marks]* Show that the language L consisting of all infinite bitstrings except $000\ldots$ (the bitstring consisting only of zeroes) is **not** PL-definable. You may want to prove the following lemma in order to solve this question:

Lemma:

For every PL-definable language L and bitstring $x \notin L$ there exists a finite prefix y of x such that for any infinite bitstring w, $yw \notin L$ (yw refers to the concatenation of y and w).

(b) *[5 + 5 points]* Show that PL-definable languages are closed neither under countable union nor under complementation.

Hint: Try using the result proven in part (a)

(c) *[10 points]* Show that a PL-definable language either contains every bitstring or does not contain uncountably many bitstrings.

Hint: Try using the lemma proven in part (a)

(d) *[Bonus 10 points]* A student tries to extend the definition of PL-languages by allowing the use of "dummy" variables.

Let $\mathbf{X} = \{x_0, x_1, \ldots\}$ denote a countably infinite set of "dummy" variables and let Σ denote a countable set of formulae over both positional and dummy variables. An infinite bitstring w is in the language defined by Σ if and only if there exists an assignment $\alpha : \mathbf{P} \cup \mathbf{X} \to \{0,1\}$ such that $\alpha \models \Sigma$ and $w_i = \alpha(p_i)$ for each i. Note that the assignment of "dummy" variables in $\mathbf{X}$ are not represented in w. Let us call the languages definable this way extended PL-definable languages, or EPL-definable languages.

Show that EPL and PL are equally expressive, ie every EPL-definable language is a PL definable language and vice versa. This means our attempt to strengthen PL this way has failed. You can use the following theorem without proof:

Theorem:

Let $S_0, S_1, S_2, \ldots$ denote an infinite sequence of non-empty sets of finite bitstrings such that for every $i > 0$ and for every bitstring $x \in S_i$ and every $j \leq i$, there exists a prefix y of x in S_j. Then there exists an infinite bitstring z such that every S_i contains a prefix of z.

Solution:

(a) Let us first prove the lemma mentioned in the question.

Lemma:

For every PL-definable language L and bitstring $x \notin L$ there exists a finite prefix y of x such that for any infinite bitstring w, $yw \notin L$ (yw refers to the concatenation of y

and w).

Proof:

Say $L = L(\Sigma)$ for some countable set of formulae Σ. If $x \notin L(\Sigma)$, then there must be some $F \in \Sigma$ such that $x \not\models F$. Since F is a formula, and all formulas are finite strings by definition, F contains only a finite number of positional variables $p \in \mathbf{P}$. Let the largest index of any positional variable present in F be n. For any infinite bitstring z with first $n+1$ bits being the same as x, the values of all the positional variables present in F are the same in both x and z, which means that $z \not\models F$ as well. This means that, if we let y denote the finite prefix of x consisting of the first $n+1$ bits of x, then for any infinite bitstring w, $yw \not\models F$, by the same argument. This means that $yw \notin L$, proving the lemma. In fact, by a similar argument, the lemma can be strengthened to stating that for any infinite bitstring $x \notin L$, there exists a finite set of positions S, such that for any infinite bitstring y, such that bits of y at the positions in S match those of x, $y \notin L$ as well.

Now, consider L as defined in the problem, ie consisting of every infinite bitstring, except $000\ldots$. Let $x = 000\ldots$. We will show that L is not PL-definable, by contradiction. Assume L is PL-definable. By the lemma we just proved, there exists a finite prefix y of x such that for any infinite bitstring w, $yw \notin L$, ie there are infinitely many bitstrings not in L. This contradicts the fact that $000\ldots$ is the only bitstring not in L. Therefore, L cannot be defined in PL.

(b) Consider the language $L = \{000\ldots\}$ (this language consists only of the infinite bitstring $000\ldots$). This language can be defined in PL as $L(\Sigma)$ where $\Sigma = \{\neg p_0, \neg p_1, \neg p_2 \ldots\}$. However, its complement is the language consisting of all infinite bitstrings other than $000\ldots$, which, as shown earlier, is not PL-definable. Therefore, PL-definable languages are not closed under complementation.

Consider the countably infinite family of languages $L_i = L(\{p_i\})$ for each $i \in \mathbb{N}$. Each L_i consists of strings where the i^{th} bit is 1, and clearly, each L_i is PL-definable. Let $L = \bigcup_{i=0}^{\infty} L_i$. L is the language consisting of all infinite bitstrings other than $000\ldots$, which, as shown earlier, is not PL-definable. Therefore, PL-definable languages are not closed under countable union.

(c) This follows directly from the lemma that was proven earlier. If a PL-definable language does not contain an infinite bitstring x, then there exists a finite prefix y of x such that for every infinite bitstring w, yw is not in the language. Since there are uncountably many infinite bitstrings w, there are uncountably many infinite bitstrings not in the language

(d) Firstly, it is easy to see that every PL-definable is also EPL-definable: PL is a special case of EPL where no dummy variables are used.

We will now show that every EPL-definable language is PL-definable. Say $\Sigma = \{F_0, F_1, \ldots\}$ is a countable set of formulae over the variables in $\mathbf{P} \cup \mathbf{X}$.

Consider $\Sigma' = \{\bigwedge_{j=0}^{i} F_j : i \in \mathbb{N}\} = \{F_0, F_0 \wedge F_1, F_0 \wedge F_1 \wedge F_2, \ldots\}$. We will show that $L(\Sigma) = L(\Sigma')$.

For any word w, $w \in L(\Sigma)$ if and only if there exists an assignment $\alpha : \mathbf{P} \cup \mathbf{X} \to \{0,1\}$ such that $w_i = \alpha(p_i)$ for each natural i and for each $F \in \Sigma$, $\alpha \models F$, ie $\alpha \models F_0$, $\alpha \models F_1$, and so on. This is equivalent to saying $\alpha \models F_0$, $\alpha \models F_0 \wedge F_1$, $\alpha \models F_0 \wedge F_1 \wedge F_2$, so on, ie $w \in L(\Sigma')$. Therefore, $L(\Sigma) \subseteq L(\Sigma')$. In a similar manner, if $w \in L(\Sigma')$, then there cannot be any F_i such that the corresponding assignment $\alpha \not\models F_i$. Hence, $w \models \Sigma'$, and $L(\Sigma) \subseteq L(\Sigma')$. From the two inclusions proved above, we have $L(\Sigma) = L(\Sigma')$. We will henceforth work with Σ', and denote $F_0 \wedge F_1 \cdots \wedge F_i$ as F_i'. F_i' satisfy a special property - for any assignment α, if $\alpha \models F_i'$, then for every $j \leq i$, $\alpha \models F_j'$.

Some Notation:

- Let $Vars_d(F)$ denote the set of dummy variables whose indices are at most the largest index of a dummy variable in the formula F - for example, if $F = p_0 \vee x_0 \vee x_2$, then $Vars_d(F) = \{x_0, x_1, x_2\}$. We have $Vars_d(F_i') \subseteq Vars_d(F_{i+1}')$ for every natural i.
- For any EPL formula F, let $Ass_d(F)$ denote the set of possible assignments to the set $Vars_d(F)$ of dummy variables
- For any formula F and assignment α to the dummy variables in F, let $F(\alpha)$ denote the formula obtained by substituting each dummy variable x with its value $\alpha(x)$. Note that $F(\alpha)$ no longer contains any dummy variables and only has positional variables, ie it is a formula in PL.

Define $\Sigma'' = \{ \bigvee_{\alpha \in Ass_d(F_i')} F_i'(\alpha) : F_i' \in \Sigma' \}$. This is a countable set of PL-formulae. We will show that $L(\Sigma'') = L(\Sigma')$, which will imply that $L(\Sigma'') = L(\Sigma)$.

Say some word $w \in L(\Sigma')$. This means there exists some assignment $\alpha : \mathbf{P} \cup \mathbf{X} \to \{0,1\}$ such that $\alpha(p_i) = w_i$ and $\alpha \models F_i'$ for each natural i. Now, since $\alpha \models F_i'$, we have $\alpha \models F_i'(\alpha)$, which implies that $\alpha \models \bigvee_{\alpha \in Ass_d(F_i')} F_i'(\alpha)$ for each natural i, which means $w \in L(\Sigma'')$.

On the other hand, say $w \in L(\Sigma'')$. This means that $w \models \bigvee_{\alpha \in Ass(Vars_d(F_i'))} F_i'(\alpha)$ for each i, ie for each i, there exists an assignment $\alpha_i : Vars_d(F_i') \to \{0,1\}$ such that $w \models F_i'(\alpha)$. Let S_i denote the set of such assignments, interpreted as finite bitstrings (eg: $x_0 \to 0, x_1 \to 1, x_2 \to 0$ is interpreted as the bitstring 010). Now, for any $\alpha \in S_i$, $w \models F_i'(\alpha)$, which means that for any $j \leq i$, $w \models F_j'(\alpha)$ as well. This means there is a prefix of the bitstring corresponding to α in each S_j, for each $j \leq i$. Therefore, the theorem can be applied, and hence there is an infinite bitstring such that every S_i contains a prefix of it. This infinite bitstring denotes an assignment to the entire set of dummy variables, and hence there exists an assignment $\alpha : \mathbf{X} \to \{0,1\}$ such that $w \models F_i'(\alpha)$ for each i, ie there exists an assignment $\alpha' : \mathbf{P} \cup \mathbf{X} \to \{0,1\}$ such that $\alpha'(p_i) = w_i$ and $\alpha'(x_i) = \alpha(x_i)$ for each i, and, as we have seen, such an α will have $\alpha \models F_i'$ for each i. Therefore, $w \in L(\Sigma')$. This means that $w \in L(\Sigma'')$ if and only if $w \in L(\Sigma')$, and hence $L(\Sigma'') = L(\Sigma') = L(\Sigma)$.

Therefore, every EPL-definable language can also be defined in PL.