Logic in CS Autumn 2024

Problem Sheet 2

S. Krishna

1. An adequate set of connectives is a set such that for every formula there is an equivalent formula with only connectives from that set. For example, $\{\neg, \vee\}$ is adequate for propositional logic since any occurrence of $\wedge$ and $\rightarrow$ can be removed using the equivalences

$$\varphi \rightarrow \psi \equiv \neg\varphi \vee \psi$$

$$\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$$

   (a) Show that $\{\neg, \wedge\}$, $\{\neg, \rightarrow\}$ and $\{\rightarrow, \bot\}$ are adequate sets of connectives. ($\bot$ treated as a nullary connective).

   (b) Show that if $C \subseteq \{\neg, \wedge, \vee, \rightarrow, \bot\}$ is adequate, then $\neg \in C$ or $\bot \in C$.

Solution

We assume the set of all connectives is $\{\top, \bot, \neg, \wedge, \vee, \rightarrow\}$.
To show that a subset of these connectives is adequate, we need to prove that for every formula constructed from the original set of connectives there is an equivalent formula constructed from the connectives in the subset. This can be proven via structural induction over the formulae.

(a) To show that $\{\neg, \wedge\}$ is adequate, we proceed by structural induction.

Base Case:

We have $\top \equiv \neg(p \wedge \neg p)$, $\bot \equiv p \wedge \neg p$, and $p \equiv p$

Inductive Steps:

Say φ and ψ are formulae constructed from the original set of connectives that have equivalent formulae that use only $\neg$ and $\wedge$. Let these equivalent formulae be φ' and ψ' respectively.

We shall show that every formula constructed from φ and ψ have equivalent formulae that use only $\neg$ and $\wedge$.

- $\neg\varphi \equiv \neg\varphi'$
- $\varphi \wedge \psi \equiv \varphi' \wedge \psi'$
- $\varphi \vee \psi \equiv \neg(\neg\varphi' \wedge \neg\psi')$
- $\varphi \rightarrow \psi \equiv \neg(\varphi' \wedge \neg\psi')$

Since φ' and ψ' contain only $\neg$ and $\wedge$, the formulae on the RHS also contain only $\neg$ and ψ', and therefore the induction step is completed. Therefore, every formula can be rewritten into an equivalent formula that uses only $\neg$ and $\wedge$.

To show that $\{\neg, \to\}$ and $\{\to, \bot\}$ are also adequate sets we follow a similiar method. The equivalences that we will use are:

i. For $\{\neg, \to\}$:
- $\top \equiv p \to p$
- $\bot \equiv \neg(p \to p)$
- $\varphi \wedge \psi \equiv \neg(\varphi \to \neg\psi)$
- $\varphi \vee \psi \equiv (\neg\varphi \to \psi)$

ii. For $\{\to, \bot\}$:
- $\top \equiv \bot \to p$
- $\neg\varphi \equiv (\varphi \to \bot)$
- $\varphi \wedge \psi \equiv ([\varphi \to (\psi \to \bot)] \to \bot)$
- $\varphi \vee \psi \equiv ([\varphi \to \bot] \to \psi)$

(b) We shall show that for any $C \subseteq \{\top, \bot, \neg, \wedge, \vee, \to\}$, if $\bot \notin C$ and $\neg \notin C$, then C cannot be adequate (This is equivalent to proving that if C is adequate then it contains either $\neg$ or $\bot$).

Before this, notice that if $C \subseteq C' \subseteq \{\top, \bot, \neg, \wedge, \vee, \to\}$ and if C is adequate, then clearly C' is adequate too. So we shall prove the above statement by showing that $\{\top, \wedge, \vee, \to\}$ is not adequate. We shall do this by showing that no formula made out of these connectives is equivalent to $\bot$.

Lemma. *For any formula made out of* $\{\top, \wedge, \vee, \to\}$*, setting all the propositional variables to* 1 *always results in the overall formula having a truth value of* 1*.*

Note that this immediately shows that no formula constructed only out of $\{\wedge, \vee, \to\}$ can be equivalent to $\bot$.

We shall prove this lemma via structural induction.

Base Case:

If the formula just consists of a single propositional variable p or is just $\top$, the result clearly follows.

Inductive Step:

Say φ and ψ are formulae constructed only with $\{\top, \wedge, \vee, \to\}$ and setting all the propositional variables to 1 results in the truth values of both φ and ψ being 1. The formulae we can construct from φ and ψ are $\varphi \wedge \psi$, $\varphi \vee \psi$ and $\varphi \to \psi$. If we set all propositional variables to 1, then by the inductive hypothesis, the truth value of φ and ψ also become 1, and it can be seen that the truth values of the new formulae are also 1.

Therefore, by structural induction, the lemma is proven and with it we have proved that $\{\top, \wedge, \vee, \to\}$ is inadequate.

2. The binary connective `nand`, $F \downarrow G$, is defined by the truth table corresponding to $\neg(F \wedge G)$. Show that `nand` is complete - that is, it can express all binary Boolean connectives.

Solution

The following equivalences (which can be verified via truth tables) imply that $\downarrow$ (`nand`) is complete, ie it can express all binary connectives.

(a) $p \wedge q \equiv (p \downarrow q) \downarrow (p \downarrow q)$

(b) $p \vee q \equiv (p \downarrow p) \downarrow (q \downarrow q)$

(c) $p \rightarrow q \equiv p \downarrow (q \downarrow q)$

$\downarrow$ can also express the unary and nullary connectives ($\neg$ and $\bot$) respectively:

(a) $\neg p \equiv p \downarrow p$

(b) $\top \equiv p \downarrow (p \downarrow p)$

(c) $\bot \equiv (p \downarrow (p \downarrow p)) \downarrow (p \downarrow (p \downarrow p))$

We can show, via structural induction, that a subset of connectives can express all connectives (not just binary ones) iff it is adequate.

Follow-up Question. Similar to `nand` is another binary connective, known as `nor`, $F \uparrow G$ with truth table, same as that of $\neg(F \vee G)$. Can you show that `nor` is also complete?

3. The binary connective `xor`, $F \oplus G$ is defined by the truth table corresponding to $(\neg F \wedge G) \vee (F \wedge \neg G)$. Show that `xor` is not complete - that is, it cannot express all binary Boolean connectives.

Solution

The truth table for $\oplus$ is as follows:

p	q	$p \oplus q$
0	0	0
0	1	1
1	0	1
1	1	0

We will show by structural induction that any formula φ constructed from two propositional variables (say p, q) will have an even number of 1s in its truth table. This means a formula like $p \wedge q$ that has an odd number of 1s in its truth table cannot be expressed via $\oplus$, ie $\oplus$ is not complete[a].

Base Case:

$\top$, $\bot$, p and q all have an even number of 1s in their truth tables.

Inductive Step:

Say φ and ψ are formulae formed with $\oplus$. By the inductive hypothesis, they have an even number of 1s in their truth tables. Let's say φ and ψ are both 0 in i places, are

both 1 in j places, φ is 0 and ψ is 1 in k places and φ is 1 while ψ is 0 in l places. By the inductive hypothesis, $j+k$ and $j+l$ are even. Therefore, their sum, $2j+k+l$ is even, which means $k+l$ is also even. By truth table, $\varphi \oplus \psi$ is 1 in $k+l$ places, and therefore its truth table also has an even number of 1s.
Therefore, any formula formed this way has an even number of 1s, and hence $\oplus$ is not complete.

Follow-up Question. As homework, you can show that all formulae whose truth tables contain an even number of 1s can be expressed with $\{\top, \bot, \oplus\}$.

[a] We will prove something stronger, as our proof will also allow $\top$, $\bot$ to be used - we prove $\{\oplus, \top, \bot\}$ is neither complete nor adequate

4. If a contradiction can be derived from a set of formulae, then the set of formulae is said to be inconsistent. Otherwise, the set of formulae is consistent. Let $\mathcal{F}$ be a set of formulae. Show that $\mathcal{F}$ is consistent iff it is satisfiable.

Solution.

First, we will show that satisfiablility implies consistency, ie if $\mathcal{F}$ is satisfiable, then it is consistent ($\mathcal{F} \nvdash \bot$).
Assume, this was not the case and there exists an assignment α such that $\alpha \models \mathcal{F}$ and $\mathcal{F} \vdash \bot$. Since our Formal Proof System is sound[a], we have $\mathcal{F} \models \bot$, and therefore, we have an assignment α such that $\alpha \models \bot$, which is not possible! Therefore, satisfiablility implies consistency.
Now, we will show the reverse, ie if $\mathcal{F}$ is consistent ($\mathcal{F} \nvdash \bot$) then it must be satisfiable. Since our Formal Proof System is complete[b], we have $\mathcal{F} \nvDash \bot$, ie there exists an assignment α such that $\alpha \models \mathcal{F}$ and $\alpha \nvDash \bot$. The latter is always true, but the former shows that $\mathcal{F}$ is satisfiable.

[a] The proof of this can be found here
[b] The proof of this can be found here

5. Suppose $\mathcal{F}$ is an inconsistent set of formulae. For each $G \in \mathcal{F}$, let $\mathcal{F}_G$ be the set obtained by removing G from $\mathcal{F}$.
   (a) Prove that for any $G \in \mathcal{F}$, $\mathcal{F}_G \vdash \neg G$, using the previous question.
   (b) Prove this using a formal proof.

Solution

We know that $\mathcal{F}$ is inconsistent, ie $\mathcal{F} \vdash \bot$

(a) By the previous result, $\mathcal{F}$ must be unsatisfiable, ie for all assignments α, $\alpha \nvDash \mathcal{F}$. Now, $\mathcal{F}$ can be written as $\mathcal{F}_G \cup \{G\}$, ie $\forall \alpha, \alpha \nvDash \mathcal{F}_G \cup \{G\}$, which can be rewritten

as $\forall\alpha, \neg(\alpha \models \mathcal{F}_G) \vee \alpha \not\models G$. This can be further rewritten as $\forall\alpha, \alpha \models \mathcal{F}_G \implies \alpha \models \neg G$, which is the definition of $\mathcal{F}_G \models \neg G$. Now, since the Formal Proof System is complete, this also means that $\mathcal{F}_G \vdash \neg G$.

(b) i. $\mathcal{F}_G \cup \{G\} \vdash \bot$ (Premise)

ii. $\mathcal{F}_G \vdash \neg G$ ($\neg$ introduction on (i))

6. Consider a formula φ which is of the form $C_1 \wedge C_2 \wedge \ldots C_n$ where each clause C_i is of the form $(\top \rightarrow \alpha)$ or $(\alpha_1 \wedge \ldots \alpha_n \rightarrow \beta)$ or $(\gamma \rightarrow \bot)$ where $\alpha, \alpha_i, \beta, \gamma$ are literals. A logician wishes to apply `HornSAT` to this formula φ by renaming negative literals (if any) with fresh positive literals. Thus, if any $\alpha, \alpha_i, \beta, \gamma$ was of the form $\neg p$, the logician will replace that $\neg p$ with a fresh variable p'. The logician claims that he can check satisfiability of φ correctly by applying `HornSAT` on the new formula (call it φ') in the following way: φ is satisfiable iff `HornSAT` concludes that φ' is satisfiable, and φ is unsatisfiable iff `HornSAT` concludes that φ' is unsatisfiable. Do you agree with the logician?

Solution

Consider $\varphi = (p \rightarrow \neg p) \wedge (\neg p \rightarrow p)$. This formula is unsatisfiable. However, after renaming, the formula becomes $(p \rightarrow p') \wedge (p' \rightarrow p)$, which is satisfiable. Therefore, the logician is incorrect.

Note that the renaming process introduces new variables which do not retain the original relationship with the existing variables. In this example, p' is treated as an independent variable after renaming and does not inherently represent the negation of the existing variable p. This lack of representation can lead to incorrect conclusions about the satisfiability of the original formula, as the new formula φ does not fully capture the logical dependencies present in φ.

7. We have seen in class that `HornSAT` has a polynomial satisfiability, while general `SAT` is NP-complete. Here is a reduction called "Hornification" proposed by a student from `SAT` to `HornSAT`. Given a formula φ in CNF, "hornify" each non-horn clause as follows.

   - If we have a clause $C_1 = p \vee q \vee r$, then all occurrences of p, q are renamed to $\neg p'$ and $\neg q'$ where p', q' are fresh variables (In general, you could have chosen to rename all but one positive literal to Hornify). Clearly, this renaming can be done in polynomial time.
   - Additionally, to respect the relationship between the original variables and their renamed counterparts, add a new Horn clause $p' \wedge p \rightarrow \bot$ (and similarly for q') whenever you rename p as $\neg p'$. This ensures that the new variables p' and q' behave correctly with respect to their original negated forms.

   Call the new formula (on an expanded set of variables) as φ'. Since φ' is in `HornSAT`, we can check its satisfiability in polynomial time. Can we conclude that "Hornification" makes `SAT` to be in P?

Solution

Notice that both the steps of "Hornification" process can be done in polynomial time. So, if we take a CNF formula, "hornify" it and check its satisfiability using `HornSAT`, the algorithm runs in polynomial time, but the algorithm may not be correct, i.e, "Hornification" does not necessarily preserve the satisfiability of the original formula, φ.

Let $\varphi = (p \vee q \vee r) \wedge (\neg p) \wedge (\neg q) \wedge (\neg r)$. Clearly, φ is UNSAT. Now after "Hornification" we get, $\varphi' = (\neg p' \vee \neg q' \vee r) \wedge (\neg p) \wedge (\neg q) \wedge (\neg r) \wedge (p' \wedge p \to \bot) \wedge (q' \wedge q \to \bot)$. But, $p = 0, p' = 0, q = 0, q' = 0, r = 0$ is a satisfying assignment to φ'. Thus $\texttt{HornSAT}(\varphi)$ will return SAT.

Notice that the main difficulty is that we are not able to rightly capture the relationship between original and renamed variables even by introducing new Horn clauses. In fact, if we rename p by $\neg p'$, we need the equivalence $p \leftrightarrow \neg p'$ to hold. But, this cannot be expressed as a Horn clause. To see why, we can rewrite this as $p \leftrightarrow \neg p' \equiv (p \to \neg p') \wedge (\neg p' \to p) \equiv (\neg p \vee \neg p') \wedge (p \vee p')$. Observe that the last clause, $p \vee p'$, can't be written as Horn clause. This means that "Hornification" cannot accurately represent the negation relationship, leading to potentially incorrect satisfiability results.

8. Using resolution, show that $P_1 \wedge P_2 \wedge P_3$ is a consequence of

$$F := (\neg P_1 \vee P_2) \wedge (\neg P_2 \vee P_3) \wedge (P_1 \vee \neg P_3) \wedge (P_1 \vee P_2 \vee P_3)\,.$$

Solution

The negation of $P_1 \wedge P_2 \wedge P_3$ is equivalent to $\neg P_1 \vee \neg P_2 \vee \neg P_3$. The following is a resolution refutation of $F \wedge (\neg P_1 \vee \neg P_2 \vee \neg P_3)$ in which clauses are represented as sets of literals:

$$\dfrac{\dfrac{\dfrac{\{P_1, P_2, P_3\} \quad \{P_1, \neg P_3\}}{\{P_1, P_2\}} \quad \{\neg P_1, P_2\}}{\{P_2\}} \quad \dfrac{\dfrac{\dfrac{\{\neg P_1, \neg P_2, \neg P_3\} \quad \{\neg P_1, P_2\}}{\{\neg P_1, \neg P_3\}} \quad \{P_1, \neg P_3\}}{\{\neg P_3\}} \quad \{\neg P_2, P_3\}}{\{\neg P_2\}}}{\emptyset}$$

9. Show that the satisfiability of any 2-CNF formula can be checked in polynomial time.

Solution

We will show that we can run resolution in time polynomial in the size of the 2-CNF. Say the CNF has k clauses in n variables. Note that performing resolution on any two clauses of the CNF will again give a clause with at most two literals. Therefore,

during the entire resolution process, we will only be dealing with clauses containing at most two literals.

Now the total possible number of such clauses is $\binom{2n}{2} + 2n$ (since there are $2n$ possible literals). The total number of possible resolutions that can be done is therefore $\binom{\binom{2n}{2}+2n}{2}$ which is $\Theta(n^4)$. Therefore, we will perform $O(n^4)$ resolutions, and since the remaining steps of performing the resolution and checking whether a clause has already been seen can be done in polynomial time, our algorithm will terminate in polynomial time. A more sophisticated analysis may reveal tighter bounds.

10. Call a set of formulae minimal unsatisfiable iff it is unsatisfiable, but every proper subset is satisfiable. Show that there exist minimal unsatisfiable sets of formulae of size n for each $n \geq 1$.

Solution

It can be easily shown that the set

$$\Sigma_n = \{p_1, \ldots p_n, \bigvee_{i=1}^{n} \neg p_i\}$$

is an example of a minimal unsatisfiable set for $n \geq 1$. The proof is left to you as an exercise.