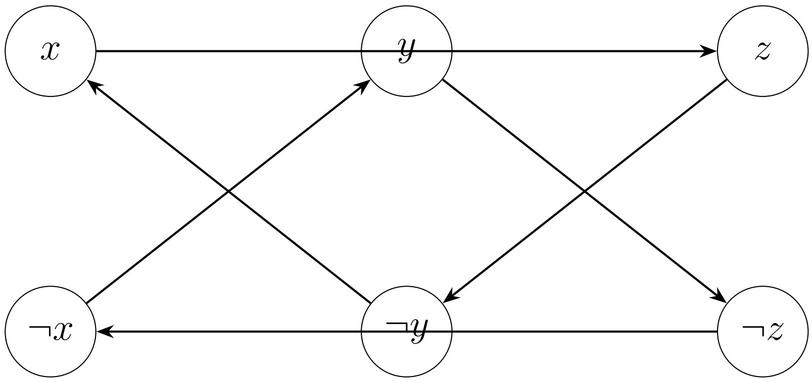
CS 228M : Logic for Computer Science Spring 2026

Practice Problem Set 2 Solutions

---

1. **2-CNF Graphs**

Solution:



Suppose that in the implication graph of a 2-CNF formula, a variable x and its negation $\neg x$ lie in the same strongly connected component.

By the definition of a strongly connected component, there exists a directed path from x to $\neg x$ and a directed path from $\neg x$ to x.

Consider any truth assignment.

- If x is assigned `true`, then following the directed path from x to $\neg x$, the implications represented by the edges force $\neg x$ to be `true`, which is a contradiction.
- If x is assigned `false`, then $\neg x$ is `true`. Following the directed path from $\neg x$ to x, the implications force x to be `true`, again a contradiction.

In both cases, a contradiction arises. Therefore, no satisfying assignment exists, and the formula is unsatisfiable.

2. **Exponential Blow-up***

Solution: Consider the family $\mathcal{F} = \{\varphi_n \mid n \in \mathbb{N}\}$, where the variables are

$$X_1, Y_1, X_2, Y_2, \ldots, X_n, Y_n$$

and the DNF formula φ_n is defined as:

$$\varphi_n := (X_1 \wedge Y_1) \ \vee \ (X_2 \wedge Y_2) \ \vee \ \cdots \ \vee \ (X_n \wedge Y_n).$$

(a) φ_n is in DNF. Each term $(X_i \wedge Y_i)$ is a cube, and φ_n is a disjunction of cubes, hence it is a DNF. Also it has 2n variables which is O(n).

(b) Length of φ_n is $O(n)$. There are n cubes, each containing exactly 2 literals. Therefore,

$$\text{length}(\varphi_n) = 2n \in O(n).$$

(c) Any equivalent CNF must have exponential length. Let ψ_n be any CNF formula over the same variables that is semantically equivalent to φ_n.

Claim 1. Every clause C in ψ_n must contain at least one of X_i or Y_i for each $i \in \{1, 2, \ldots, n\}$.

Proof. Suppose for contradiction that there exists a clause C and an index i such that neither X_i nor Y_i appears in C. Consider the assignment α defined as:

$$\alpha(X_i) = 1, \quad \alpha(Y_i) = 1,$$

and for every literal appearing in C, set its truth value so that C becomes false under α (this is always possible since C does not contain X_i or Y_i). Then φ_n evaluates to 1 under α (because $X_i \wedge Y_i$ is true), but ψ_n evaluates to 0 (because clause C is false). This contradicts $\psi_n \equiv \varphi_n$.

Thus, every clause C in ψ_n must pick at least one literal from each pair $\{X_i, Y_i\}$. Hence each clause corresponds to a choice of one literal from each pair, i.e., a tuple in the Cartesian product:

$$\{X_1, Y_1\} \times \{X_2, Y_2\} \times \cdots \times \{X_n, Y_n\}.$$

Claim 2. For every tuple $(L_1, L_2, \ldots, L_n)$ where $L_i \in \{X_i, Y_i\}$, there must exist a clause C in ψ_n that contains all literals $L_1, L_2, \ldots, L_n$.

Proof. Suppose not. Then there exists a tuple $(L_1, \ldots, L_n)$ such that no clause in ψ_n contains all of $L_1, \ldots, L_n$. Construct an assignment β as follows:

$$\beta(L_i) = 1 \text{ for all } i = 1, \ldots, n,$$

and set the complementary literal in each pair to 0 (i.e., if $L_i = X_i$ then set $Y_i = 0$, and if $L_i = Y_i$ then set $X_i = 0$). Under this assignment, for every i, at least one of X_i, Y_i is 0, so $(X_i \wedge Y_i)$ is false for all i, and hence

$$\varphi_n(\beta) = 0.$$

Now consider any clause C of ψ_n. By Claim 1, C contains at least one of X_i or Y_i for each i. But since C does not contain the entire tuple $(L_1, \ldots, L_n)$, there exists some index j such that C contains the other literal in the pair (not L_j), which is false under β. Thus every clause C has at least one true literal under β, so $\psi_n(\beta) = 1$, contradicting $\psi_n \equiv \varphi_n$.

There are exactly 2^n distinct tuples in

$$\{X_1, Y_1\} \times \cdots \times \{X_n, Y_n\}.$$

By Claim 2, ψ_n must contain at least one clause per tuple, so ψ_n has at least 2^n clauses. Each such clause contains at least n literals (one from each pair), hence:

$$\text{length}(\psi_n) \;\geq\; n \cdot 2^n,$$

which is exponential in n. Therefore, no semantically equivalent CNF formula of polynomial length exists.

Solution:

1. Let $C = \{p, l_1, \cdots, l_n\} \in F$ be a blocked clause over the literal p. Consider the set of clauses $\{D_1, \cdots, D_m\} \in F$ containing $\neg p$, then for all $D_i s$, there exists at least one literal $\neg l_j \in D_i$. Now, to show that F and $F \backslash \{C\}$ are equisatisfiable. We show that if $F \backslash \{C\}$ is unsatisfiable F is unsatisfiable (trivially) and if $F \backslash \{C\}$ is satisfiable F is also satisfiable. We will show this by considering the following cases over the complete assignments α:

   (a) there exists an α satisfying $F \backslash \{C\}$ such that p is set to 1 in α: As p is set to 1, C is also true.

   (b) All the α that satisfy $F \backslash \{C\}$ have $p := 0$: This means that there is atleast one clause D_i, whose all other literals are set to 0. Thus, some literal $l_i \in C$ would have become 1 and thus C is true.

   Therefore, F and $F \backslash \{C\}$ equisatisfiable.

2. Consider the CNF formula

$$\varphi = (x \vee y) \ \wedge \ (\neg x \vee \neg y).$$

   The clause $C = (x \vee y)$ is a blocked clause in φ. Removing C yields the formula

$$\varphi' = (\neg x \vee \neg y).$$

   Let $\alpha : \{x, y\} \to \{\top, \bot\}$ be the total assignment defined by

$$\alpha(x) = \bot, \quad \alpha(y) = \bot.$$

   Then $\alpha \models \varphi'$, but $\alpha \not\models \varphi$.

3. **Substitution Theorem**

Solution: We prove by structural induction on H. If H is atomic, the only subformula of H is H itself. So $F = H$. It follows that $H' = G$ and, since $F \equiv G$, we have $H \equiv H'$.

The induction hypothesis is that the theorem holds for formulas H_1 and H_2 (structurally having lesser connectives than H) each of which contains an occurrence of F as a subformula. That is, $H_1 \equiv H_1'$ and $H_2 \equiv H_2'$ whenever H_1' and H_2' are formulas obtained from H_1 and H_2 by replacing an occurrence of F with G.

Suppose $H = \neg H_1$. Then $H' = \neg H_1'$ Since $H_1 \equiv H_1'$, we have $\neg H_1 \equiv \neg H_1'$. It follows that $H \equiv H'$ as was required.

Suppose H is one of the following formulas: $H_1 \wedge H_2, H_1 \vee H_2$ or $H_1 \to H_2$. Since F is a subformula of H, F is a subformula of H_1, a subformula of H_2, or is H itself. If $F = H$, then we have $H = F \equiv G = H'$ as in the atomic case. So we may assume that the occurrence of F that is to be replaced by G occurs either in H_1 or H_2. Without loss of generality, assume that it occurs in H_1.

If $H = H_1 \wedge H_2$ then $H' = H_1' \wedge H_2$. In this case we have: $H_1 \wedge H_2$ is true if and only if both H_1 and H_2 are true if and only if both H_1' and H_2 are true (since $H_1 \equiv H_1'$) if and only if $H_1' \wedge H_2$ is true. That is, $H_1 \wedge H_2 \equiv H_1' \wedge H_2$. Since $H \equiv H_1 \wedge H_2$, we have

$H \equiv H'$.

If $H = H_1 \vee H_2$, then $H' = H_1' \vee H_2$. By the definition of $\vee$, we have $H \equiv \neg(\neg H_1 \wedge \neg H_2)$ and $H' \equiv \neg(\neg H_1' \wedge H_2)$. It follows from the previous cases (corresponding to $\neg$ and $\wedge$) that $H \equiv H'$.

If $H = H_1 \to H_2$, then $H' = H_1' \to H_2$. By the definition of $\to$, $H \equiv (\neg H_1 \vee H_2$ and $H' \equiv (\neg H_1' \vee H_2)$. It follows from the previous cases (corresponding to $\neg$ and $\vee$) that $H \equiv H'$.

Therefore for any formula H that contains F as a subformula, $H \equiv H'$.

4. **Linear Resolution***

Solution: Let F be a set of formulae which is unsatisfiable. Let $F' \subseteq F$ be a minimal set of unsatisfiable formulae (unsatisfiable prime). The claim is that starting with any formula in F', we can linearly resolve to get ϕ. We prove by induction on the number of atomic formulae in F'. Let C be a clause in F'. Then we have two cases: $|C| = 1$ or $|C| > 1$.

Case 1: $|C| = 1$
If $C = \{l\}$ where l is a literal, WLOG assume l is positve and $l = p$. Then $F'[p/1]$ is unsatisfiable with an unsatisfiable core F''. There is a clause C' in F'' such that $C' \cup \{\neg l\} \in F'$ (otherwise $F'' \subseteq F' - \{C\}$ so F'' is satisfiable by minimality of F'). Now by the induction hypothesis, create a linear resolution - $(*)$ of F'' to get the empty clause.
Then start with resolving C and $C' \cup \{\neg l\}$ and append the derivation $(*)$ to get $\neg l$. Finally, resolve with C to get ϕ.

Case 2: $|C| > 1$
Choose a literal $l \in C$. Let $C' = C - \{l\}$. Then $F'[l/0]$ is unsatisfiable so it has an unsatisfiable core F'' which also contains C'. If not, $F'[l/0] - \{C'\} = (F' - \{C\})[l/0]$ is unsatisfiable. But $F'[l/1] = (F' - \{C\})[l/1]$ is also unsatisfiable. Therefore $F' - \{C\}$ is unsatisfiable, contradicting minimality of F'.
Then construct a linear resolution of the empty clause from F'' starting with C' by induction hypothesis. Transform that into a resolution of l from F' starting with C. Having a derived l, use the previous case to derive the empty clause from $(F' - \{C\}) \cup \{\{l\}\}$ starting with l. This is an unsatisfiable core because $(F' - \{C\})$ is satisfiable.

5. **k-SAT to 3-SAT**

Solution: Let

$$G = (\ell_1 \vee \ell_2 \vee \cdots \vee \ell_k), \qquad k \geq 4.$$

Using the standard Tseitin-style chain encoding, introduce fresh auxiliary variables

$$t_1, t_2, \ldots, t_{k-3}$$

that do not appear in the original formula, and define the 3-CNF formula

$$G' \ = \ (\ell_1 \vee \ell_2 \vee t_1) \ \wedge \ \bigwedge_{i=1}^{k-4} (\neg t_i \vee \ell_{i+2} \vee t_{i+1}) \ \wedge \ (\neg t_{k-3} \vee \ell_{k-1} \vee \ell_k).$$

Each clause has exactly three literals, hence G' is in 3-CNF.

Linear blow-up. The construction introduces $k-3$ new variables (namely $t_1, \ldots, t_{k-3}$). The number of clauses in G' is

$$1 \;+\; (k-4) \;+\; 1 \;=\; k-2,$$

which is linear in k. Hence the increase in both variables and clauses is $O(k)$.

6. **Polytime Satisfiability**

Solution:

- Process variables one by one. If a variable x appears 0 or 1 time, remove it (if it comes once, set it to satisfy that single occurrence).
- If x appears twice with the same polarity, assign x to satisfy that polarity and delete the satisfied clauses.
- If x appears once positively in $C_1 = A \vee x$ and once negatively in $C_2 = B \vee \neg x$, replace C_1, C_2 in F by their resolvent $A \vee B$, lets call if F' (in original resolution we don't remove original clauses since we may have other clauses with that variable). It follows from resolution that value that satisfy F also satisfy F'. We show that assignment satisfying F' also satisfy F. Since this assignment satisfy the resolvent it satisfy either A or B. In the former case setting x to false would satisfy original formula and in latter case setting x to true would satisfy original formula. So basically we can keep eliminating variables without affecting satisfiability.
- Repeat until either an empty clause appears (UNSAT) or no clauses remain (SAT).

7. **K true satisfiability***

Solution:
Define $\Psi := F \wedge C_{\leq k}(x_1, \ldots, x_n) \wedge C_{\leq n-k}(\neg x_1, \ldots, \neg x_n)$, where $C_{\leq t}(y_1, \ldots, y_n)$ is a CNF encoding the constraint "at most t of the literals y_i are true" by the sequential counter: Introduce fresh variables $s_{i,j}$ for $1 \leq i \leq n-1,\ 1 \leq j \leq t$. Intuitively $s_{i,j}$ means "among $y_1, \ldots, y_i$ there are at least j true". The clauses are:

$$\begin{aligned}
&(\neg y_1 \vee s_{1,1}),\\
&(\neg s_{i-1,j} \vee s_{i,j}) \quad \text{for } 2 \leq i \leq n-1,\ 1 \leq j \leq t,\\
&(\neg y_i \vee \neg s_{i-1,j-1} \vee s_{i,j}) \quad \text{for } 2 \leq i \leq n-1,\ 2 \leq j \leq t,\\
&(\neg y_i \vee s_{i,1}) \quad \text{for } 2 \leq i \leq n-1,\\
&(\neg y_n \vee \neg s_{n-1,t}) \quad \text{(overflow clause).}
\end{aligned}$$

Apply this with $t = k$ and $y_i = x_i$, and with $t = n-k$ and $y_i = \neg x_i$. The total number of new variables and clauses is $O(nk)$.

Correctness (sketch).

($\Rightarrow$) If Ψ is satisfiable then its restriction to X satisfies F. The two encodings enforce "at most k of the x_i are true" and "at least k of the x_i are true" respectively, hence exactly k.

($\Leftarrow$) If α is a satisfying assignment of F with exactly k true x_i's, extend α to the $s_{i,j}$ by setting $s_{i,j}$ = true iff among the first i relevant literals there are at least j true. Then every sequential-counter clause is satisfied (monotonicity and overflow match the intended counts), so Ψ is satisfiable.

Size. The encoding uses $(n-1)\cdot t$ auxiliaries and $O(nt)$ clauses for a bound t; applied twice (with $t = k$ and $t = n-k$) the size is $O(n \cdot \max\{k, n-k\})$ = polynomial in n, k. In particular one can achieve $O(nk)$.

8. **Positive Resolution***

Solution: We prove the desired result by induction on the number of variables in F. The base case is that F has no variables. Then we must have $F = \{\}$, so there is a trivial positive resolution refutation of F.

The induction step is as follows. Pick a variable P in F and consider the formulas $F_0 := F[\textbf{false}/P]$ (notation replace P by true/false) and $F_1 := F[\textbf{true}/P]$. Since F is unsatisfiable, F_0 and F_1 are both unsatisfiable. Since F_0 contains one fewer variable than F, by the induction hypothesis there is a positive resolution proof $C_0, C_1, \ldots, C_m = \bot$ that derives the empty clause from F_0. For each clause C_i in the above proof that comes from F_0, either C_i is already in F or $C_i \cup \{P\}$ is in F. By replacing the deleted copies of P from the latter type of clause we obtain a new (still positive) resolution proof $C'_0, C'_1, \ldots, C'_m$ from F, where either C'_m is empty or $C'_m = \{P\}$. In the first case there is nothing more to prove, so suppose $C'_m = \{P\}$, i.e., we have a positive resolution derivation of $\{P\}$ from F.

Now we consider F_1. By induction, we also have a positive resolution proof $C_0, \ldots, C_m = \{\}$ from F_1. For each clause C_i in the above proof that comes from F_1, either C_i is already in F or $C_i \cup \{\neg P\}$ is in F. In the latter case we can obtain C_i from $C_i \cup \{\neg P\}$ and $\{P\}$ by a single positive resolution step. Thus we can construct a positive resolution proof from F.

9. **HornSAT**

Solution:

(a) The given Horn formula is

$$F = (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_3 \vee x_4) \wedge (\neg x_4) \wedge (\neg x_5 \vee x_3) \wedge (\neg x_6 \vee x_5) \wedge (\neg x_7 \vee x_6).$$

Since there are no clauses of the form $\top \to x$, we don't require setting variables to 1.

After this there are no clauses of the form $l_1 \wedge l_2 \wedge \ldots l_n \to \bot$. Thus the formula is satisifable. The assignment assigning everything false is a satisfying assignment.

(b) After the Horn-SAT algorithm terminates, the assignment obtained is the *minimal model* of the formula.

(i) **Example where assigning additional variables to 1 works.**
Consider the Horn formula:

$$G = (\neg x \vee y).$$

The Horn-SAT algorithm does not force any variable to be 1, so the minimal model is:

$$x = 0, \quad y = 0.$$

If we now assign:

$$x = 1, \quad y = 1,$$

the clause $(\neg x \vee y)$ is still satisfied. Hence, assigning additional variables to 1 preserves satisfiability in this case.

(ii) **Example where assigning additional variables to 1 fails.**
Consider the Horn formula:

$$H = (\neg x) \wedge (\neg y \vee x).$$

Since there are no clauses of the form $\top \to x$, we don't require setting variables to 1.

Now if we assign 1 to x, it clearly is a non satisfying assignment.

10. **Hyper Resolution**

Solution: Applying Hyper Resolution on following set of clauses:

$$(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_5) \wedge (\neg x_2 \vee x_5) \wedge (\neg x_3 \vee x_5)$$

results in a unit clause x_5.
Applying Hyper Resolution on following set of clauses :

$$(x_6 \vee x_8) \wedge (\neg x_6 \vee \neg x_5) \wedge (\neg x_8 \vee \neg x_5)$$

results in a unit clause $\neg x_5$.
Resolving the two unit clauses by propositional resolution, we get unsatisfiable. Observe that hyper resolution combines multiple resolution steps into one.