Proof Systems for Propositional Logic

Mahesh Viswanathan

Fall 2018

If logic is the science of valid inference, then proofs embody its heart. But what are mathematical proofs? They are a sequence of statements where each statement in the sequence is either a self evident truth, or "logically" follows from previous observations. Thus, sound derivation principles are identified by correct proofs.

Example 1. Euclid's Elements sets out axioms (or postulates), which are self evident truths, and proves all results in geometry from these truths formally. Euclid lays out five axioms for geometry.

A1 A straight line can be drawn from any point to any point.

A2 A finite line segment can be extended to an infinite straight line.

A3 A circle can be drawn with any point as center and any given radius.

A4 All right angles are equal.

A5 If a straight line falling on two straight lines makes the interior angles on the same side less than two right angles, the straight lines, if produced indefinitely, will meet on that side on which the angles are less than two right angles.

Using these axioms, Euclid proves a number of results in geometry. He uses previously proved propositions in the proofs of later observations. An example of such a result, is the proof that the sum of the interior angles of a triangle is 180°.

Proposition 2. *The interior angles of a triangle sum to two right angles.*

Proof. Consider the diagram in Figure 1b. The proposition is proved using the following sequence of statements.

1. Extend one side (say) BC to D [A2]
2. Draw a line parallel to AB through point C; call it CE [P31]
3. Since AB is parallel to CE, BAC = ACE and ABC = ECD [P29]

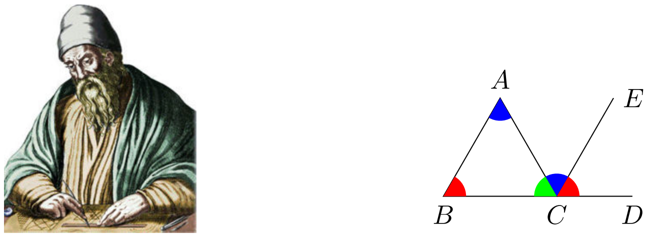

Figure 1: (a) Euclid of Alexandria (left); (b) Sum of the internal angles of a triangle (right).

$$\frac{}{\varphi \to (\psi \to \varphi)} \qquad \frac{}{(\varphi \to (\psi \to \rho)) \to ((\varphi \to \psi) \to (\varphi \to \rho))} \qquad \frac{}{((\varphi \to \bot) \to \bot) \to \varphi}$$

$$\frac{\varphi \qquad \varphi \to \psi}{\psi}$$

Figure 2: A Frege-style Proof System

4. Thus, the sum of the interior angles = ACB + ACE + ECD = 180°

References [P31] and [P29] in steps 2 and 3, allude to previously propositions 31 and 29, proved in the book. □

Example 1 highlights the basic elements of identifying good proofs — one needs to identify axioms, and the principle by which new conclusions can be drawn from previously established facts. A formal proof system for a logic identifies such *axioms* and *rules of inference*. We will introduce two such proof systems for propositional logic — a Frege-style proof system, and resolution — to give a flavor of different types of proof systems.

1 A Frege-style Proof System

Proof systems are most convenient presented as a collection of rules of the form

$$\frac{\Gamma}{\varphi}$$

where Γ is a set of formulas (schemas) and φ is a formula (schema). Such rules can be interpreted as follows — if every formula in Γ can be established then φ can be concluded from these observations. One special case is when $\Gamma = \emptyset$. In this case the formula below the line can be concluded without establishing anything; in other words, it is an axiom. Instead of explicitly writing $\emptyset$ above the line, we simply don't write anything, and present this axiom in the form

$$\frac{}{\varphi}$$

Our first proof system is shown in Figure 2. The first row shows the 3 axiom schemas in our proof system, while the second row shows the only rule of inference. The formulas φ, ψ, and ρ in Figure 2, can be *any formulas*. For example, taking $\varphi = p$ and $\psi = p$, we get $p \to (p \to p)$ as an *instantiation* of the first axiom schema, while taking $\varphi = p$ and $\psi = p \to p$, we get $p \to ((p \to p) \to p)$ as a different instantiation of the *same* schema. The rule of inference in this proof system, is a very commonly used rule. It, therefore, has a special name; it is called *modus ponens*.

Proofs in our (formal) proof system, will be like the usual proofs in mathematics — they will be a sequence of statements. However, instead of using english statements, here they will simply be well formed formulas of propositional logic. The statement or formulas being proved is the last one in the sequence. The sequence of formulas in a proof should be consistent with the axioms and rule of inference of the proof system, for it to be *valid* proof. This is captured in the definition below.

Definition 3 (Proofs)**.** A *proof* of φ from a set (possibly infinite) of hypotheses Γ is a finite sequence of wffs $\psi_1, \psi_2, \ldots \psi_m$ such that $\psi_m = \varphi$, and for every $k \in \{1, 2, \ldots m\}$, either

- $\psi_k \in \Gamma$, or
- ψ_k is an axiom, or

- ψ_k follows from ψ_i and ψ_j, with $i, j < k$, by modus ponens.

The *length* of such a proof is the number of wffs in the sequence, namely, m. If there is a proof of φ from Γ, we denote this by $\Gamma \vdash \varphi$. When $\Gamma = \emptyset$, we write this as $\vdash \varphi$ (as opposed to $\emptyset \vdash \varphi$).

Let us look at some proofs in our system.

Example 4. Let us construct a proof of $q \to p$ from the hypothesis $\{p\}$. Such a proof is as follows.

1.	$p \to (q \to p)$	Axiom 1, taking $\varphi = p$, and $\psi = q$
2.	p	Hypothesis in set Γ
3.	$q \to p$	Modus Ponens on lines 1 and 2

Thus, $\{p\} \vdash q \to p$.

We will now show that $\vdash \bot \to ((p \to q) \to (p \to p))$.

1.	$(p \to (q \to p)) \to ((p \to q) \to (p \to p))$	Axiom 2, taking $\varphi = p$, $\psi = q$ and $\rho = p$
2.	$p \to (q \to p)$	Axiom 1, taking $\varphi = p$, and $\psi = q$
3.	$(p \to q) \to (p \to p)$	Modus ponens on lines 1 and 2
4.	$((p \to q) \to (p \to p)) \to (\bot \to ((p \to q) \to (p \to p)))$	Axiom 1, taking $\varphi = (p \to q) \to (p \to p)$, and $\psi = \bot$
5.	$\bot \to ((p \to q) \to (p \to p))$	Modus ponens on lines 3 and 4

Finally, let us show $\vdash p \to p$.

1.	$(p \to ((p \to p) \to p)) \to ((p \to (p \to p)) \to (p \to p))$	Axiom of 2, taking $\varphi = p$, $\psi = p \to p$ and $\rho = p$
2.	$(p \to ((p \to p) \to p))$	Axiom 1, taking $\varphi = p$, and $\psi = p \to p$
3.	$(p \to (p \to p)) \to (p \to p)$	Modus ponens on lines 1 and 2
4.	$p \to (p \to p)$	Axiom 1, taking $\varphi = p$, $\psi = p$
5.	$p \to p$	Modus ponens on lines 3 and 4

In proof systems, like the one we are considering in this section, there is a very useful theorem that makes writing proofs easy. This is called the *deduction theorem*. Some proof systems have it as an explicit rule.

Theorem 5 (Deduction Theorem). *If* $\Gamma \cup \{\varphi\} \vdash \psi$ *then* $\Gamma \vdash \varphi \to \psi$.

First, observe that the converse of Theorem 5, is clearly true, i.e., if $\Gamma \vdash \varphi \to \psi$ then $\Gamma \cup \{\varphi\} \vdash \psi$. Establishing this left as an exercise. The proof of the deduction theorem is a more difficult exercise. The informal outline of the proof is as follows. Assume that $\rho_1, \rho_2, \ldots \rho_m$ is a proof of ψ from $\Gamma \cup \{\varphi\}$. One shows by induction on i that, for each line i, we have $\Gamma \vdash \varphi \to \rho_i$.

The deduction theorem simplifies the task of writing down proofs in our proof system.

Example 6. Consider the task of showing $\vdash (\varphi \to \psi) \to ((\psi \to \rho) \to (\varphi \to \rho))$, where φ, ψ, and ρ are arbitrary wffs. Our approach to solving this problem, would instead be to instead establish $\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \rho$; we succeed, we will get the desired result by using the deduction theorem a few times.

1.	$\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \varphi \to \psi$	Hypothesis
2.	$\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \varphi$	Hypothesis
3.	$\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \psi$	Modus Ponens on lines 1 and 2
4.	$\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \psi \to \rho$	Hypothesis
5.	$\{\varphi \to \psi, \psi \to \rho, \varphi\} \vdash \rho$	Modus Ponens on lines 3 and 4
6.	$\{\varphi \to \psi, \psi \to \rho\} \vdash \varphi \to \rho$	Deduction Theorem
7.	$\{\varphi \to \psi\} \vdash (\psi \to \rho) \to (\varphi \to \rho)$	Deduction Theorem
8.	$\vdash (\varphi \to \psi) \to ((\psi \to \rho) \to (\varphi \to \rho))$	Deduction Theorem

1.1 Completeness Theorem

Proofs in a formal proof system like the one above try capture the notion of correct logical inference. But what do we mean by "correct logical inference"? There are two aspects to such a question. First, any conclusion drawn by a proof must be logically correct, i.e., consistent with the semantics we defined. In other words, we should not be able to conclude any false facts using a proof. This is often refered to as the *soundness* of the proof system. The second is that the proof system must be rich enough to be able to prove all true facts. This is called the *completeness* of the proof system. We make this connection between provability and the semantics we gave, precise in the following theorem.

Theorem 7 (Soundness and Completeness)**.** *For any set of formulas Γ (possibly even infinite) and any wff φ the following two properties hold.*

Soundness *If $\Gamma \vdash \varphi$ then $\Gamma \models \varphi$.*

Completeness *If $\Gamma \models \varphi$ then $\Gamma \vdash \varphi$.*

Thus, any formula proved without any hypotheses is a tautology, and every tautology has a proof from the empty set of hypotheses in our proof system. We will not prove Theorem 7. We will instead prove such a soundness and completeness theorem for the second proof system that we will introduce for propositional logic. Proving soundess is usually easy. It requires making sure that the axioms and proof rules are consistent with the semantics of the logic. In this case it requires showing that every axiom in the proof system is indeed a tautology, and modus ponens is consistent with logical consequence. Proving completeness is typically hard.

2 Resolution

Notice that the proofs for formulas that we constructed in Examples 4 and 6 are very symbolic and mechanical — one doesn't need to understand what the formula we are trying to prove is saying or what the meaning of the hypotheses is. Instead the proofs are constructed by looking at the pattern of formulas. This raises the prospect of trying to mechanize the process of searching for proofs of formulas. However, the proof system in Section 1 is not good for this purpose. This is because at any point during the construction of the proof, one can extend it by using any one of the axiom schemas. Since each axiom schema can be instantiated in infinitely many possible ways, this makes mechanization difficult. A proof system that is amenable to mechanization is one that has very few choices at any step of the proof construction. *Resolution* is such a proof system. Remarkably, resolution (for propositional logic) has no axioms and only one rule of inference.

Resolution works when the formulas are represented in *conjunctive normal form* (CNF). We begin, therefore, by recalling the definition of conjunctive normal form formulas. To define CNF formulas, it is convenient to think of propositional logic formulas as being built up using $\neg$, $\wedge$, and $\vee$, instead of using $\rightarrow$ and $\perp$. It will also be convenient to think of $\vee$ and $\wedge$ being present implicitly (see Definition 8 below) instead of explictly in the syntax. Hopefully, the reader will not be confused with these relaxations to the syntax of propositional logic we are making.

Definition 8 (Conjunctive Normal Form)**.** Conjunctive normal form formulas are defined as follows.

- A *literal* is a proposition p or its negation $\neg p$.
- A *clause* is a disjunction of literals. We will think of a clause as a set of literals, implicitly assuming that the literals are disjuncted. In this interpretation, a truth valuation satisfies a clause if some literal in the set evaluates to 1 under the truth valuation.
- The *empty clause* is the clause containing no literals. By definition, no truth assignment satisfies the empty clause.

- A formula is said to be in *conjunctive normal form* (CNF) if it is conjunction of clauses. Again, we will think of a CNF formula as a set of clauses, with the conjunction being implicit in the syntax. With this interpretation, a truth valuation satisfies a CNF formula if it satisfies every clause in the set representing the formula.

CNF formulas are formulas in a restricted form. However, they are not semantically restrictive. That is, every wff φ can be shown to be equivalent to a formula ψ in CNF — we can push negations all the way in using DeMorgan's Laws, and then distribute AND over ORs. Let us look at some examples of CNF and non CNF formulas.

Example 9. The formulas $(p_1 \wedge q_1) \vee (p_2 \wedge q_2)$, $\neg(p \wedge q)$ are examples of formulas that are not in CNF; the first formula does not have $\wedge$ as the topmost connective, while the second formula does not have negations pushed all the way inside. The formulas $(p_1 \vee p_2) \wedge (p_1 \vee q_2) \wedge (q_1 \vee p_2) \wedge (q_1 \vee q_2)$ and $\neg p \vee \neg q$ are formulas in CNF as they are conjunction of clauses.

We will represent CNF formulas as set of set of literals, without explict conjunctions and disjunctions. For example, $\{\{p_1, p_2\}, \{p_1, q_2\}, \{q_1, p_2\}, \{q_1, q_2\}\}$ is the way the formula $(p_1 \vee p_2) \wedge (p_1 \vee q_2) \wedge (q_1 \vee p_2) \wedge (q_1 \vee q_2)$ will be represented. Similarly, $\{\{\neg p, \neg q\}\}$ will be the representation of $\neg p \vee \neg q$.

The resolution proof system is a sequence of transformations that preserve satisfiability, until the empty clause (which is by definition not satisfiable) is obtained. The transformations involve the single rule of inference that constructs the *resolvent* of two clauses.

Definition 10 (Resolvent)**.** The only rule in the resolution proof system is as follows.

$$\frac{C \cup \{p\} \qquad D \cup \{\neg p\}}{C \cup D}$$

The conclusion $C \cup D$ is called the *resolvent* of $C \cup \{p\}$ and $D \cup \{\neg p\}$ with respect to proposition p.

Example 11. Consider the clauses $\{p, \neg q, \neg r\}$ and the clause $\{\neg p, \neg q\}$. The resolvent of these two clauses (with respect to p) is the clause $\{\neg q, \neg r\}$.

Definition 12 (Refutations)**.** A *resolution refutation* of a (possibly infinite) set of clauses Γ is a sequence of clauses $C_1, C_2, \ldots C_m$ such that each clause C_k is either in Γ or a resolvent of two clauses C_i and C_j $(i, j < k)$, and the last clause C_m in the refutation is the empty clause.

Example 13. The set of clauses $\Gamma = \{\{p, q\}, \{\neg p, r\}, \{\neg q, r\}, \{\neg r\}\}$ has the following resolution refutation.

1.	$\{\neg p, r\}$	
2.	$\{\neg r\}$	
3.	$\{\neg p\}$	Resolvent of 1 and 2
4.	$\{\neg q, r\}$	
5.	$\{\neg q\}$	Resolvent of 2 and 4
6.	$\{p, q\}$	
7.	$\{q\}$	Resolvent of 3 and 6
8.	$\{\}$	Resolvent of 5,7

2.1 Proving Tautologies with Resolution

The resolution proof system differs from the Frege-style proof system that we introduced in the previous section in one aspect. The proofs in the proof system from Section 1 are designed to establish tautologies. In resolution, on the other hand, we construct *refutations*, which as the name suggests are designed to prove the *unsatisfiability* of a formula given in CNF; recall that a formula φ is unsatisfiable if $\mathsf{v}[\![\varphi]\!] = 0$ for all valuations v. Establishing unsatisfiability and validity are closely related.

Proposition 14. *A formula φ is a tautology if and only if $\neg\varphi$ is unsatisfiable.*

Proof. The proposition can be established by the following sequence of observations. φ is a tautology iff for every valuation v, $\mathsf{v}[\![\varphi]\!] = 1$ (definition of tautology) iff for every valuation v, $\mathsf{v}[\![\neg\varphi]\!] = 0$ (from the semantics of $\neg$) iff $\neg\varphi$ is unsatisfiable (definition of unsatisfiability). □

Since resolution refutations aim to establish the unsatisfiability of a set of clauses, to prove that a formula is a tautology, we need to use Proposition 14. That is, to prove that φ is a tautology, we need to convert $\neg\varphi$ into CNF. This can be done as follows.

1. Express $\psi_1 \to \psi_2$ as $\neg\psi_1 \vee \psi_2$.
2. Push negations inside using DeMorgan's Laws. Recall that DeMorgan's laws say the following.
$$\neg(\psi_1 \wedge \psi_2) \equiv \neg\psi_1 \vee \neg\psi_2 \qquad \neg(\psi_1 \vee \psi_2) \equiv \neg\psi_1 \wedge \neg\psi_2$$
3. Remove double negations, because $\neg\neg\psi \equiv \psi$
4. Distribute conjunction over disjunction, using the distributive law, which says
$$\psi_1 \vee (\psi_2 \wedge \psi_3) \equiv (\psi_1 \vee \psi_2) \wedge (\psi_1 \vee \psi_3) \qquad (\psi_1 \wedge \psi_2) \vee \psi_3 \equiv (\psi_1 \vee \psi_3) \wedge (\psi_2 \vee \psi_3)$$

Example 15. Let $\varphi = (\neg p_1 \vee \neg q_1) \wedge (\neg p_2 \vee \neg q_2)$. We can convert $\neg\varphi$ to CNF as follows.

1. Pushing negations inside using DeMorgan's Laws, we get
$$(\neg\neg p_1 \wedge \neg\neg q_1) \vee (\neg\neg p_2 \wedge \neg\neg q_2)$$
2. Removing double negations, we get
$$(p_1 \wedge q_1) \vee (p_2 \wedge q_2)$$
3. Distributing conjunctions over disjunctions, we get
$$(p_1 \vee p_2) \wedge (p_1 \vee q_2) \wedge (q_1 \vee p_2) \wedge (q_1 \vee q_2)$$

The above method while semantically correct, can be expensive. The main reason is that when we distribute conjunctions over disjunctions, the resulting formula can be exponentially larger. For example, if we have the formula
$$\bigvee_{i=1}^{n} (p_i \wedge q_i)$$
and we distribute conjuctions over disjunctions, we will get a CNF formula where each clause is of the form $(r_1 \vee r_2 \vee \cdots \vee r_n)$, where r_i is either p_i or q_i. This will result in a formula with 2^n clauses (as we have two choices for each r_i).

The exponential blowup can be avoided by using a translation proposed by Tsejtin. Tseijtin's method does not construct a logically equivalent CNF formula. Instead, for the formula $\neg\varphi$, it constructs a CNF formula ψ with the property that $\neg\varphi$ is satisfiable if and only if ψ is satisfiable. This weaker correspondence between $\neg\varphi$ and ψ is sufficient in this context; we will have φ is a tautology if and only if ψ is unsatisfiable.

Tsejtin's Method. Let us describe the conversion of formula $\neg\varphi$. The first step is to introduce new *extension* propositions x_ψ for each subformula ψ of φ as follows.

- For each proposition in φ, the extension proposition x_p is the same as p.
- For each negated subformula $\neg\psi$, $x_{\neg\psi}$ is taken to be the literal $\neg x_\psi$.
- for all other subformulas ψ, x_ψ is a new proposition.

Having identified the extension propositions, the CNF formula that we will construct corresponding to $\neg\varphi$ is as follows. Here, we will use the representation of CNF formulas as sets of sets of literals. So for φ, we define Γ_φ to be the following set of clauses.

- The singleton clause $\{\neg x_\varphi\}$
- For each subformula $\psi \wedge \rho$, we add the CNF formula equivalent to "$x_{\psi\wedge\rho} \leftrightarrow x_\psi \wedge x_\rho$. In other words, we will have the clauses

$$\{\neg x_{\psi\wedge\rho}, x_\psi\} \qquad \{\neg x_{\psi\wedge\rho}, x_\rho\} \qquad \{x_{\psi\wedge\rho}, \neg x_\psi, \neg x_\rho\}$$

- For each subformula $\psi \vee \rho$, we add the CNF formula equivalent to "$x_{\psi\vee\rho} \leftrightarrow x_\psi \vee x_\rho$. In other words, we will have the clauses

$$\{x_{\psi\vee\rho}, \neg x_\psi\} \qquad \{x_{\psi\vee\rho}, \neg x_\rho\} \qquad \{\neg x_{\psi\vee\rho}, x_\psi, x_\rho\}$$

Observe that the number of subformulas of a given formula φ is linear in the size of φ. Thus the number of extension propositions we introduce is linear in φ. Further, for each subformula, we are introducing only a constant (3 to be precise) number of clauses. This means that resulting set of clauses Γ_φ is linear in the size of φ.

Example 16. Let us apply Tsejtin's construction to the formula in Example 15. Recall that $\varphi = (\neg p_1 \vee \neg q_1) \wedge (\neg p_2 \vee \neg q_2)$. The first step is to identify the new extension propositions we need. In this case there are only 3 that we will add — x_φ corresponding to φ, x_1 corresponding to $(\neg p_1 \vee \neg q_1)$, and x_2 corresponding to $(\neg p_2 \vee \neg q_2)$. Our CNF formula Γ_φ will be obtained by adding 3 clauses for each of these interesting subformulas corresponding to x_φ, x_1 and x_2. Thus, we have Γ_φ is the following set

$$\left\{\begin{array}{l} \{\neg x_\varphi\}, \\ \{\neg x_\varphi, x_1\}, \{\neg x_\varphi, x_2\}, \{x_\varphi, \neg x_1, \neg x_2\}, \\ \{x_1, p_1\}, \{x_1, q_1\}, \{\neg x_1, \neg p_1, \neg q_1\}, \\ \{x_2, p_2\}, \{x_2, q_2\}, \{\neg x_2, \neg p_2, \neg q_2\} \end{array}\right\}$$

In the above description, we have replaced $\neg\neg p$ by p for $p \in \{p_1, q_1, p_2, q_2\}$.

2.2 Completeness of Resolution

We will now prove that resolution is a "correct" proof system. In other words, we will prove the soundness and completeness of resolution. What that means for resolution is that a set of clauses Γ has a resolution refutation if and only if Γ is unsatisfiable. Note that we will establish this for any set Γ, including those that are infinite. We begin by recalling what satisfiability and unsatisfiability means in this context, before presenting the soundness and completeness theorems.

Definition 17. We define satisfiability of clauses (which are sets of literals) and CNF formulas (which are sets of clauses).

- A clause C is *satisfiable* if there is some truth valuation v and a literal $\ell \in C$ such that $\mathsf{v}[\![\ell]\!] = 1$.
- A set of clauses Γ is *satisfiable* if there is some valuation v such that for every clause $C \in \Gamma$, there is some literal $\ell \in C$, with $\mathsf{v}[\![\ell]\!] = 1$.

A set of clause Γ (or a clause C) is said to be *unsatisfiable*, if it is not satisfiable.

Theorem 18 (Soundness)**.** *If a set of clauses Γ has a resolution refutation, then Γ is unsatisfiable.*

Proof. The crux of the proof of the soundness theorem, is to establish the "correctness" of the resolution proof rule. That is capture by the following lemma.

Lemma 19. *Let C be the resolvent of two clauses D and E with respect proposition p. If a truth valuation v satisfies both D and E then v satisfies C.*

Proof of Lemma 19. Without loss of generality, let us assume $p \in D$ and $\neg p \in E$. Since v satisfies both D and E, there must be literals $\ell_1 \in D$ and $\ell_2 \in E$ such that $\mathsf{v}[\![\ell_1]\!] = \mathsf{v}[\![\ell_2]\!] = 1$. Observe that either $\ell_1 \neq p$ or $\ell_2 \neq \neg p$, because v can only satisfy one out of p and $\neg p$. Assume without loss of generality $\ell_1 \neq p$. Then $\ell_1 \in C$. Therefore, v satisfies C. □

Using Lemma 19, we are ready to complete the proof of Theorem 18. Let $C_1, C_2, \ldots C_m$ be a resolution refutation of Γ. Let us define a sequence of sets of clauses inductively as follows.

$$\Gamma_0 = \Gamma \qquad \Gamma_i = \Gamma_{i-1} \cup \{C_i\}$$

Thus, $\Gamma_i = \Gamma \cup \{C_1, C_2, \ldots C_i\}$. We can argue that, for every k, if Γ_{k-1} is satisfiable then so is Γ_k. This is because (a) if $C_k \in \Gamma$ then $\Gamma_k = \Gamma_{k-1}$ and the observation follows immediately; (b) on the other hand, if C_k is the resolvent of C_i and C_j $(i, j < k)$, then since Γ_{k-1} is satisfiable, there is a truth assignment say v that satisfies Γ_{k-1} (and hence both C_i and C_j) which also satisfies C_k (by Lemma 19).

Now, observe that Γ_m is unsatisfiable because it contains the $C_m = \{\}$. Therefore, because of the observations in the previous paragraph, $\Gamma = \Gamma_0$ is also unsatisfiable. □

Theorem 20 (Completeness)**.** *If a set of clauses Γ is unsatisfiable then there is a resolution refutation of Γ.*

We will present a proof of the completeness theorem due to David and Putnam. For finite Γ, the proof is constructive. That is, when Γ is unsatisfiable, it gives a specific construction of a refutation for Γ.

Proof. Let assume that Γ is a finite set. The case when Γ is infinite will be handled later. When Γ is a finite set, we will prove the completeness theorem by induction on the number of proposition appearing in Γ.

For the base case, observe that if Γ contains no propositions, then Γ contains the empty clause [1]. Then the refutation for Γ is simply $\{\}$.

Let us now consider the induction step. Let p be a proposition that appears in Γ. With respect to proposition p, Γ can be partitioned into 3 sets.

$$\begin{aligned}
\Gamma_0^p &= \{C \in \Gamma \mid C \cap \{p, \neg p\} = \emptyset\} \\
\Gamma_+^p &= \{C \in \Gamma \mid p \in C\} \\
\Gamma_-^p &= \{C \in \Gamma \mid \neg p \in C\}
\end{aligned}$$

Thus, Γ_0^p are clauses where p does not appear, Γ_+^p are those where p appears positively, and Γ_-^p are those where p appears negatively. Let us construct a new set of clauses as follows.

$$\Gamma_p = \Gamma_0^p \cup \{C \cup D \mid C \cup \{p\} \in \Gamma_+^p \text{ and } D \cup \{\neg p\} \in \Gamma_-^p\}$$

Thus, Γ_p has all the clauses in Γ_0^p and all resolvent of clauses from Γ_+^p and Γ_-^p. Observe that p no longer appears in Γ_p. If we can argue that Γ_p is unsatisfiable then we can complete the proof by using the induction hypothesis — the refutation for Γ is just all the steps to creat Γ_p followed by the refutation for Γ_p.

To finish the proof in the finite case, we need to establish the following lemma.

Lemma 21. *If Γ_p is satisfiable then so is Γ.*

Proof of Lemma 21. Let v be a truth assignment that satisfies Γ_p. Let v' be the truth assignment that is identical to v, except that it flips the assignment to p. Observe that since v and v' only differ on the assignment to p, they agree on all the propositions appearing in Γ_p. Therefore, v' also satisfies Γ_p.

[1] Note, if Γ is unsatisfiable, Γ must be a non-empty set of clauses. This is because, by definition, the empty set of clauses is satisfiable.

Let us assume without loss of generality, that $\mathsf{v}(p) = 1$ and $\mathsf{v}'(p) = 0$. We will show that either v or v' satisfy Γ. Observe that both v and v' satisfy Γ_0^p (because p does not appear in Γ_0^p). Also, v satisfies Γ_+^p (because all clauses in Γ_+^p have p) and v' satisfies Γ_-^p (because all clauses in Γ_-^p have $\neg p$). Now if v satisfies Γ_-^p, v satisfies Γ. Similarly, if v' satisfies Γ_+^p then v' satisfies Γ. So the problem is if neither of these hold. In that case that is a clause $C \cup \{p\} \in \Gamma_+^p$ that is not satisfied by v' and there is a clause $D \cup \{\neg p\} \in \Gamma_-^p$ that is not satisfied by v. But then their resolvent $C \cup D \in \Gamma_p$ is not satisfied by either v or v', which contradicts our assumption that both v and v' satisfy Γ_p. □

With the proof of Lemma 21, we have proved Theorem 20 when Γ is finite. When Γ is infinite, we observe that the compactness theorem, which is established in the next section, guarantees that if Γ is unsatisfiable, then there is *finite* subset Γ_0 of Γ such that Γ_0 is unsatisfiable. Using the compactness theorem, we have completed the proof of Theorem 20. □

3 The Compactness Theorem

To prove the completeness theorem for resolution (Theorem 20) in the case when Γ is an infinite set of clauses, we relied on the observation that if Γ is unsatisfiable then there is a finite subset Γ_0 of Γ that is also unsatisfiable. This is a very important observation in called the *compactness theorem*. In this section, we will look at a couple of different proofs of this theorem.

Recall that a set of formulas Γ is *satisfiable* if there is a valuation v such that for every $\varphi \in \Gamma$, $\mathsf{v} \models \varphi$ (or $\mathsf{v}[\![\varphi]\!] = 1$). We now introduce another notion called finitely satisfiable.

Definition 22 (Finitely Satisfiable)**.** A set of formulas Γ is *finitely satisfiable* iff every finite subset Γ_0 of Γ is satisfiable.

The two notions, satisfiability and finite satisfiability, are equivalent — this is the content of the *compactness theorem.*

Theorem 23 (Compactness)**.** *A set of formulas Γ is satisfiable if and only if Γ is finitely satisfiable.*

One consequence of the compactness theorem is that if Γ is unsatisfiable then there must be a finite subset of Γ that is unsatisfiable; this is exactly what we used in Theorem 20. Second, observe that if Γ is satisfiable then the satisfying assignment (say v) also satisfies every subset of Γ and therefore also every finite subset of Γ. The challenge is, therefore, in proving the converse — that finite satisfiability implies satisfiability. If Γ is a finite set, then clearly finite satisfiability implies satisfiability because Γ itself is a finite subset of Γ. So the interesting case is when Γ is infinite. We will provide a couple of very different proofs for this result.

Before looking at the proofs of the Compactness theorem, it is useful to observe that Compactness theorem can be seen as a consequence of the completeness theorem for some proof system. Imagine there was a proof for Theorem 7 that did not rely on the Compactness Theorem. That is, for the proof system introduced in Section 1, $\Gamma \models \varphi$ iff $\Gamma \vdash \varphi$. From the definition logical consequence, observe that Γ is unsatisfiable iff $\Gamma \models \bot$. Therefore, by Theorem 7, there is a proof π of $\bot$ from Γ. Take $\Gamma_0 = \Gamma \cap \pi$. Γ_0 is a finite subset. Further, π is also a proof of $\bot$ from Γ_0. Soundness of the proof system then allows us to conclude $\Gamma_0 \models \bot$, i.e., Γ_0 is unsatisfiable.

3.1 Compactness using König's Lemma

We present a simple and elegant proof of the compactness theorem that uses König's Lemma. This proof approach works only for propositional logic, and does not extend to first order logic. Let us begin by recalling König's lemma for binary trees.

A binary tree is said to have paths of *arbitrary length* if for each natural number n, there is a path in the tree whose length is $\geq n$. An infinite path in the binary tree is an infinite sequence of vertices of the tree such that successive vertices in the sequence are connected by an edge. Observe that if a binary tree has an infinite path then it also has paths of arbitrary length. This is because for every n, the prefix of the infinite

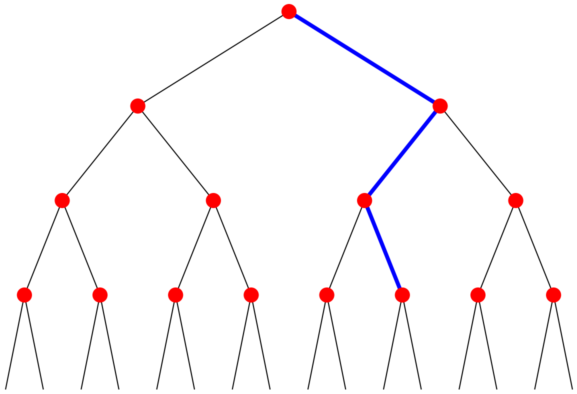

Figure 3: The blue path in the tree, corresponds to the (partial) assignment $\mathsf{v}(p_1) = 1, \mathsf{v}(p_2) = 0, \mathsf{v}(p_3) = 1$.

path with $n + 1$ vertices, is a path in the binary tree of length n. König's Lemma says that the converse of this is also true.

Lemma 24 (König)**.** *A binary tree with paths of arbitrary length has an infinite path.*

Proof. Suppose u is a vertex that does not have paths of arbitrary length starting from it, then by definition, there must be a number m such that all paths starting from u are of length at most m. Now, if a vertex u has the property that none of its children v have paths of arbitrary length starting from them, then u also cannot have paths of arbitrary length starting from it. The contrapostive of this statement is that if u is vertex with paths of arbitrary length starting from it, then at least one of its children v also has paths of arbitrary length.

Suppose a binary tree has paths of arbitrary length. Then the root is a vertex that has paths of arbitrary length starting from it. The infinite path is given by $v_0, v_1, \ldots$ where v_0 is the root, and v_{i+1} is the child of v_i that has paths of arbitrary length. □

Let us fix the set of propositions in our logic to be $\mathsf{Prop} = \{p_i \mid i \in \mathbb{N}\}$. Truth assignments to Prop can be thought of as (infinite) paths in the complete, infinite binary tree — vertices at level i correspond to proposition p_i and if the path takes the left child at level i, then it corresponds to the assignment setting p_i to 0; otherwise it sets p_i to 1. Finite paths in this tree, correspond to partial assignments. So a path of length i corresponds to an assignment that sets values to propositions $\{p_1, p_2, \ldots p_i\}$. For example, in Figure 3, the blue path corresponds to the (partial) assignment v that sets $\mathsf{v}(p_1) = 1$, $\mathsf{v}(p_2) = 0$, and $\mathsf{v}(p_3) = 1$. Recall that, whether a formula φ holds in an assignment, depends only on the truth values assigned to the propositions that appear in φ. Thus, partial assignments can determine the truth of formulas that only mention the propositions that have been assigned. For example, the partial assignment indicated by the blue path in Figure 3 can determine the truth of any formula that only mentions p_1, p_2, and p_3.

Proof of Theorem 23. Let Γ be a set of formulas that is finitely satisfiable. Logical equivalence $\equiv$ parititons Γ into equivalence classes. Take Γ' to be a subset of Γ that contains exactly one representative from each equivalence class. That is, $\Gamma' \subseteq \Gamma$ such that

- For every $\varphi \neq \psi \in \Gamma'$, $\varphi \not\equiv \psi$, and
- For every $\varphi \in \Gamma$, there is $\psi \in \Gamma'$ such that $\varphi \equiv \psi$.

Since $\Gamma' \subseteq \Gamma$, Γ' is also finitely satisfiable.

Recall that $\mathsf{prop}(\varphi)$ is defined to be the set of propositions that appear in φ. For $i \geq 0$, define Γ_i to be

$$\Gamma_i = \{\varphi \in \Gamma' \mid \mathsf{prop}(\varphi) \subseteq \{p_1, p_2, \ldots p_i\}\}.$$

Observe that Γ_i defines an non-decreasing sequence of sets, i.e., for every i, $\Gamma_i \subseteq \Gamma_{i+1}$. Also, $\Gamma' = \cup_{i\geq 0}\Gamma_i$. The most important observation about Γ_i is that it is a finite set — since Γ' has only one formula from each equivalence class of $\equiv$, each formula in Γ_i corresponds to a unique subset of assignments of $\{p_1, \ldots p_i\}$ to $\{0, 1\}$. Thus, we have $|\Gamma_i| \leq 2^{2^i}$. Since Γ' is finitely satisfiable, Γ_i is satisfiable for every i.

Define T_Γ as the following set of (partial) truth assignments.

$$T_\Gamma = \bigcup_{i\geq 1} \{\mathsf{v} : \{p_1, \ldots p_i\} \to \{0, 1\} \mid \mathsf{v} \models \Gamma_i\}$$

where we say $\mathsf{v} \models \Gamma_i$ if v satisfies all formulas in Γ_i. Consider an assignment $\mathsf{v} \in T_\Gamma$ with domain $\{p_1, p_2, \ldots p_i\}$. By definition $\mathsf{v} \models \Gamma_i$. For $j < i$, since $\Gamma_j \subseteq \Gamma_i$, $\mathsf{v} \models \Gamma_j$. Further, Γ_j only has propositions $\{p_1, \ldots p_j\}$, we also have $\mathsf{v}' \models \Gamma_j$, where v' is the restriction of v to the domain $\{p_1, \ldots p_j\}$. So $\mathsf{v}' \in T_\Gamma$. Viewed as paths in the infinite binary tree (see Figure 3), v is a path of length i, v' its prefix of length j. What we observe is that T_Γ is "closed" under prefix of paths. Thus, if we restrict our attention to the assignments in T_Γ then they form a subtree of the infinite binary tree.

Let us consider T_Γ. In the previous paragraph we observed that is forms a subtree of the infinite binary tree. It has paths of arbitrary length; this is because every Γ_i is satisfiable, and an (partial) assignment satisfying Γ_i is a path of length i in the tree. Since T_Γ is a binary tree, by König's lemma, it has an infinite path. The infinite path corresponds to a (full) truth assignment, say v_*. Further, since every prefix of v_* is a (finite) path in T_Γ, it means that the prefix of length i viewed as partial assignments satisfies Γ_i. Therefore, v_* satisfies every Γ_i, and therefore satisfies Γ'. Now, since every formula $\varphi \in \Gamma$ is logically equivalent to some formula $\psi \in \Gamma'$, it means $\mathsf{v}_* \models \Gamma$. Thus, Γ is satifiable. □

3.2 Compactness using Henkin Models

The proof of the compactness theorem we present in the section, relies on constructing a truth assignment for a set of formulas Γ through the process of *saturation*, where we add formulas to the set Γ as long as it remains finitely satisfiable. This is an approach proposed by Henkin.

Let us fix Γ to be a finitely satisfiable set of formulas. We begin by making an important observation about such sets.

Lemma 25. *Let Γ be finitely satisfiable, and let φ be any formula. It is the case that either $\Gamma \cup \{\varphi\}$ or $\Gamma \cup \{\neg\varphi\}$ is finitely satisfiable.*

Proof. Suppose $\Gamma \cup \{\varphi\}$ is not finitely satisfiable. Then there is some finite subset of $\Gamma \cup \{\varphi\}$ that is not satisfiable; let us denote such a subset by Γ_0. First observe that $\varphi \in \Gamma_0$. This is because if $\varphi \notin \Gamma_0$ then Γ_0 is a finite subset of Γ, which has to be satisfiable (since Γ is finitely satisfiable) and this contradicts our assumption on Γ_0 being unsatisfiable. Thus, we can write $\Gamma_0 = \Gamma_1 \cup \{\varphi\}$, where $\Gamma_1 \subseteq \Gamma$.

We will show that $\Gamma \cup \{\neg\varphi\}$ is finitely satisfiable. Consider an arbitrary finite subset Γ' of $\Gamma \cup \{\neg\varphi\}$. If $\neg\varphi \notin \Gamma'$ then $\Gamma' \subseteq \Gamma$ and then since Γ is finitely satisfiable, Γ' is satisfiable. The interesting case to consider is when $\neg\varphi \in \Gamma'$. In this case, $\Gamma' = \Gamma_2 \cup \{\neg\varphi\}$, where $\Gamma_2 \subseteq \Gamma$. Now, $\Gamma_1 \cup \Gamma_2$ is a finite subset of Γ, and is therefore satisfiable; let v be a truth assignment satisfying $\Gamma_1 \cup \Gamma_2$. Observe that $\mathsf{v}[\![\varphi]\!] = 0$, because otherwise we would have $\mathsf{v} \models \Gamma_1 \cup \{\varphi\}$, which contradicts our assumption that $\Gamma_1 \cup \{\varphi\}$ is unsatifiable. Therefore, $\mathsf{v} \models \Gamma_2 \cup \{\neg\varphi\}$, and so $\Gamma \cup \{\neg\varphi\}$ is finitely satisfiable. □

The set of all formulas of propositional logic are *countable*, i.e., there is a 1-to-1, onto function $f : \mathbb{N} \to \mathcal{F}$, where $\mathcal{F}$ is the set of all propositional logic formulas. Therefore, we can *enumerate* all the formulas in propositional logic. Let $\varphi_1, \varphi_2, \ldots$ be an enumeration of all formulas. Let us, inductively, define a sequence of sets of formulas as follows.

$$\begin{array}{rcl} \Delta_0 & = & \Gamma \\ \Delta_n & = & \left\{ \begin{array}{ll} \Delta_{n-1} \cup \{\varphi_n\} & \text{if this is finitely satisfiable} \\ \Delta_{n-1} \cup \{\neg\varphi_n\} & \text{otherwise} \end{array} \right. \end{array}$$

Observe that the sequence is nondecreasing, i.e., for every n, $\Delta_n \subseteq \Delta_{n+1}$. Further, by induction on n, using Lemma 25, we can conclude that Δ_n is finitely satisfiable for all n. Finally, define

$$\Delta = \bigcup_{n \in \mathbb{N}} \Delta_n.$$

Let us not look at some properties of Δ.

Proposition 26. *Δ is finitely satisfiable.*

Proof. Consider any finite subset $X = \{\psi_1, \ldots \psi_m\}$ of Δ. Observe, by definition Δ, for each i, there is some n_i such that $\psi \in \Delta_{n_i}$. Taking $n = \max\{n_1, \ldots n_m\}$, observe that $X \subseteq \Delta_n$. Since Δ_n is finitely satisfiable, X is satisfiable. This means that Δ is finitely satisfiable. □

Proposition 27. *For any formula φ, $\neg\varphi \in \Delta$ if and only if $\varphi \notin \Delta$.*

Proof. Without loss of generality assume that φ is the nth formula, i.e., $\varphi = \varphi_n$. Now by definition, in step n of the construction of Δ, if $\varphi \notin \Delta$ then $\neg\varphi \in \Delta$. On the other hand, if $\{\varphi, \neg\varphi\} \subseteq \Delta$ then since $\{\varphi, \neg\varphi\}$ is not satisfiable, Δ would not be finitely satisfiable. But since Δ is finitely satisfiable, it must be the case that at most one out of φ and $\neg\varphi$ belong to Δ. □

Proposition 28. *For any formulas φ and ψ the following properties hold.*

- *If $\varphi \equiv \psi$ then $\varphi \in \Delta$ iff $\psi \in \Delta$.*
- *$\varphi \vee \psi \in \Delta$ iff $\varphi \in \Delta$ or $\psi \in \Delta$.*
- *$\varphi \wedge \psi \in \Delta$ iff $\varphi \in \Delta$ and $\psi \in \Delta$.*

Proof. The proposition follows from Propositions 26 and 27. Details are left to the reader to flesh out. □

We are now ready to complete the proof of Theorem 23. That is, we will show that Γ (which is finitely satisfiable) is satisfiable. Consider the truth assignment v defined as follows.

$$\mathsf{v}(p) = \begin{cases} 1 & \text{if } p \in \Delta \\ 0 & \text{otherwise} \end{cases}$$

v shows that Δ is satisfiable, because of the following result.

Proposition 29. *For any formula φ, $\mathsf{v}[\![\varphi]\!] = 1$ if and only if $\varphi \in \Delta$.*

Proof. Using Propositions 27 and 28, the result is established by induction on the structure of the formula. Details are left to the reader. □

Proposition 29 establishes the fact that $\mathsf{v} \models \Delta$. Since $\Gamma \subseteq \Delta$, $\mathsf{v} \models \Gamma$. Therefore, Γ is satisfiable.

3.3 An Application of Compactness: Coloring Infinite Planar Graphs

In this section we present an application of the compactness theorem. We will show that all infinite planar graphs are 4-colorable. We begin by recalling the graph coloring problem, and its connection to propositional logic.

Definition 30 (Graphs)**.** An undirected graph $G = (V, E)$ is a set of vertices V, and a set of edges $E \subseteq V \times V$, such that E is symmetric (i.e., $(u, v) \in E$ iff $(v, u) \in E$) and irreflexive (i.e., $(u, u) \notin E$ for any $u \in V$).

Definition 31 (Coloring)**.** A k-coloring of graph $G = (V, E)$ is a function $c : V \to \{1, 2, \ldots k\}$ such that if $(u, v) \in E$ then $c(u) \neq c(v)$. If G has a k-coloring then G is said to be k-colorable.

The problem if determining whether a graph is k-colorable can be "reduced" to checking the satisfiability of a set of clauses.

Proposition 32. *For any graph $G = (V, E)$ (with possibly infinitely many vertices), there is a set of clauses $\Gamma_{G,k}$ such that G is k-colorable iff $\Gamma_{G,k}$ is satifiable.*

Proof. For each vertex $u \in V$ and $1 \leq i \leq k$, take the proposition p_{ui} to denote "vertex u has color i". $\Gamma_{G,k}$ is the following set of clauses.

- For each $u \in V$, the clause $p_{u1} \vee p_{u2} \vee \cdots \vee p_{uk}$. Intuitively these clauses capture the constraint that every vertex gets at least one of the k colors.
- For each $u \in V$ and $1 \leq i, j \leq k$ with $i \neq j$, the clause $\neg p_{ui} \vee \neg p_{uj}$. These clauses capture the constraint that a vertex does not get two different colors.
- For each edge $(u, v) \in E$ and color $1 \leq i \leq k$, the clause $\neg p_{ui} \vee \neg p_{vi}$. These clauses ensure that adjacent vertices do not get the same color.

For a coloring c, define the valuation v_c such that $\mathsf{v}_c(p_{ui}) = 1$ iff $c(u) = i$. Similarly for a valuation v, define a function $c_{\mathsf{v}}(u) = i$ iff $\mathsf{v}(p_{ui}) = 1$. Observe that

- If c is a valid k-coloring of G then v_c satisfies $\Gamma_{G,k}$, and
- If v satisfies $\Gamma_{G,k}$ then c_{v} is a valid k-coloring of G.

We leave the proof of the above observations to the reader. □

Finite, planar graphs are graphs with finitely many vertices such that there is a drawing of the graph on the plane where the edges do not cross. A celebrated result about finite, planar graphs is that 4 colors are sufficient to color the graph.

Theorem 33 (Appel-Haken). *Every finite planar graph is 4-colorable.*

We will show that the compactness theorem in fact shows that Theorem 33 can be extended to infinite graphs as well.

Corollary 34. *All infinite planar graphs are 4-colorable.*

Proof. Let G be an infinite planar graph. Consider the set of clauses $\Gamma_{G,4}$ constructed in Proposition 32. Observe that $\Gamma_{G,4}$ is finitely satisfiable. This can be seen as follows. Let Γ_0 be any finite subset of $\Gamma_{G,4}$. Let G_0 be the graph induced by the vertices u such that the proposition p_{ui} appears in Γ_0 for some i. Now, G_0 is a finite, planar graph and so by Theorem 33 has a 4-coloring c. Then by the proof of Proposition 32, the valuation v_c satisfies $\Gamma_{G_0,4}$. Since $\Gamma_0 \subseteq \Gamma_{G_0,4}$, we have v_c satisfies Γ_0.

Since $\Gamma_{G,4}$ is finitely satisfiable, by the compactness theorem, $\Gamma_{G,4}$ is satisfiable. Let v be a satisfying assignment for $\Gamma_{G,4}$. Again, by Proposition 32, c_{v} is a valid 4-coloring of G. □