Lecture 8

Propositional Proof Systems

February 3, 1998
Lecturer: Richard Ladner
Notes: Nicholas Bone

8.1. Rules of Inference

Shoenfield defined a proof system $\mathcal{S}$ based on the following five rules of inference. The formulas above the line represent an antecedent—or premise—and the formulas below the line represent the consequent—or conclusion— we may draw from these.

1. $\dfrac{}{\neg\alpha \vee \alpha}$
2. $\dfrac{\alpha}{\alpha \vee \beta}$
3. $\dfrac{\alpha \vee \alpha}{\alpha}$
4. $\dfrac{(\alpha \vee \beta) \vee \gamma}{\alpha \vee (\beta \vee \gamma)}$
5. $\dfrac{\alpha \vee \beta, \neg\alpha \vee \gamma}{\beta \vee \gamma}$ (Cut Rule)

Definitions: In the following definitions, $\mathcal{Z}$ is an arbitrary proof system and α and β are formulas.

- $\vdash_{\mathcal{Z}} \alpha$ if $\exists \beta_1, \ldots, \beta_k$, where $\beta_k = \alpha$ and $\forall i \leq k \exists i_1, \ldots, i_m < i$ such that $\dfrac{\beta_{i_1}, \ldots, \beta_{i_m}}{\beta_i}$ is an instance of a rule in $\mathcal{Z}$. In other words, there is a proof of α in $\mathcal{Z}$.
- $\models \alpha$ if α is valid (true under any assignment).
- $\mathcal{Z}$ is *sound* if $\forall \alpha, \vdash_{\mathcal{Z}} \alpha \Rightarrow \models \alpha$.
- $\mathcal{Z}$ is *complete* if $\forall \alpha, \models \alpha \Rightarrow \vdash_{\mathcal{Z}} \alpha$.

Example: The derivation of $\dfrac{\alpha, \alpha \Rightarrow \beta}{\beta}$ (Modus Ponens).

1	α	from the antecedent
2	$\neg\alpha \vee \beta$	($\equiv \alpha \Rightarrow \beta$) from the antecedent
3	$\alpha \vee \beta$	applying rule 2 to statement 1
4	$\beta \vee \beta$	applying rule 5 to statements 2 and 3
5	β	applying rule 3 to statement 4

Theorem 8.1: $\mathcal{S}$ is both sound and complete.

Proof omitted.

8.1.1. Axioms

Let Γ be a set of axioms (not necessarily valid), $\mathcal{Z}$ a proof system, and α and β formulas. Then we have the following additional definitions:

- $\Gamma \vdash_{\mathcal{Z}} \alpha$ if $\exists \beta_1, \ldots, \beta_k$, where $\beta_k = \alpha$ and $\forall i \leq k$ either $\beta_i \in \Gamma$ or $\exists i_1, \ldots, i_m < i$ such that $\dfrac{\beta_{i_1}, \ldots, \beta_{i_m}}{\beta_i}$ is an instance of a rule in $\mathcal{Z}$. In effect, Γ adds the set of rules $\{\dfrac{}{\gamma} : \gamma \in \Gamma\}$ to our proof system.
- $\Gamma \models \alpha$ if α is valid relative to Γ (any assignment which satisfies all formulas of Γ also satisfies α).
- $\mathcal{Z}$ is *strongly sound* if $\forall \Gamma \forall \alpha, \Gamma \vdash_{\mathcal{Z}} \alpha \Rightarrow \Gamma \models \alpha$.
- $\mathcal{Z}$ is *strongly complete* if $\forall \Gamma \forall \alpha, \Gamma \models \alpha \Rightarrow \Gamma \vdash_{\mathcal{Z}} \alpha$.

Theorem 8.2: $\mathcal{S}$ is strongly sound and strongly complete.

Proof omitted.

Theorem 8.3: Deduction Theorem for $\mathcal{S}$: Let Γ be a set of formulas, and α and β be formulas. Then $\Gamma \cup \{\alpha\} \vdash_{\mathcal{S}} \beta$ if and only if $\Gamma \vdash_{\mathcal{S}} \alpha \Rightarrow \beta$.

This can be proved by induction on proof length.

8.2. Algorithms for Validity

Proof systems are pretty, but they don't give insight into how to find a proof. The following are three algorithmic approaches for determining the validity of Boolean formulas.

Remark: $\models \alpha \Leftrightarrow \neg\alpha$ is not satisfiable.

8.2.1. Basic Truth Table Approach

Given a formula, α, for each possible assignment, σ, check if $\sigma \models \alpha$. This procedure takes exponential time.

8.2.2. Davis-Putnam Approach

First a notational definition: Let α be a formula and x be a variable appearing in α. Then $\alpha(x = T)$ is the formula obtained by replacing every occurrence of x in α with T, and simplifying using rules to remove T and F from formulas. Removal rules include replacing $T \vee \beta$ with T, $T \wedge \beta$ with β and so on. The expression $\alpha(x = F)$ is defined similarly.

The Davis-Putnam algorithm is defined as follows:

- $DP(T)$ =true; $DP(F)$ =false
- For a formula α of $n > 0$ variables, choose a variable x in α and return $DP(\alpha(x = T)) \wedge DP(\alpha(x = F))$. Note that the formulas $\alpha(x = T)$ and $\alpha(x = F)$ have no more than $n - 1$ variables.

This algorithm can also run in exponential time.

8.2.3. Resolution

More notation: If a is a literal, then define $\overline{a}$ to be the negation of a. I.e., if a is the assertion of the variable x, then $\overline{a}$ is the assertion of $\neg x$, and vice versa.

Disjunctive clauses $a_1 \vee a_2 \vee \cdots \vee a_n$ are represented as sets $\{a_1, a_2, \ldots, a_n\}$. A CNF-formula $\bigwedge_{i=1}^{m} \bigvee_{j=1}^{n_i} a_{ij}$ is represented as a set of sets $\{\{a_{11}, a_{12}, \ldots, a_{1n_1}\}, \ldots, \{a_{m1}, a_{m2}, \ldots, a_{mn_m}\}\}$.

A CNF-formula $\mathcal{F} = \{C_1, C_2, \ldots, C_n\}$ is satisfiable if there is a set S of literals with the following two properties:

1. $a \in S \Rightarrow \overline{a} \notin S$
2. $\forall i, S \cap C_i \neq \emptyset$

Such an S is called a *partial satisfying assignment.*

Let C and D be two clauses such that $a \in C$ and $\overline{a} \in D$. Then the *resolvent* of C and D by a is $(C - \{a\}) \cup (D - \{\overline{a}\})$ (essentially the same as Shoenfield's rule 5).

Lemma 8.4: Let $\mathcal{F}$ be a CNF-formula, and let C and D be two clauses in $\mathcal{F}$ such that $a \in C$ and $\overline{a} \in D$. Then $\mathcal{F}$ is satisfiable $\Leftrightarrow \mathcal{F} \cup \{E\}$ is satisfiable, where E is the resolvent of C and D by a.

Proof: ($\Leftarrow$) Trivial.

($\Rightarrow$) Let S be a partially satisfying assignment for $\mathcal{F}$. S cannot contain both a and $\overline{a}$. If $a \notin S$ then $(C - \{a\}) \cap S \neq \emptyset$, so $E \cap S \neq \emptyset$. If $\overline{a} \notin S$ then $(D - \{\overline{a}\}) \cap S \neq \emptyset$, so $E \cap S \neq \emptyset$. □

Let $\mathcal{F}$ be a CNF-formula. Construct a chain $\mathcal{F} = \mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \cdots$, where each $\mathcal{F}_i$ is obtained by applying resolution to two clauses of $\mathcal{F}_{i-1}$. The chain eventually terminates with some $\mathcal{F}_m$. $\mathcal{F}_m$ is called the *resolution closure* of $\mathcal{F}$, and is denoted $r(\mathcal{F})$.

Theorem 8.5: Let $\mathcal{F}$ be a CNF-formula. Then $\mathcal{F}$ is satisfiable $\Leftrightarrow \emptyset \notin r(\mathcal{F})$.

Proof: ($\Rightarrow$) Follows from 8.4 and the second condition for satisfiability.

($\Leftarrow$) We need only show that $r(\mathcal{F})$ is satisfiable; 8.4 will take care of the rest.

Assume $\emptyset \notin r(\mathcal{F})$, and let $x_1, x_2, \ldots, x_n$ be the variables of $\mathcal{F}$. Define $\tilde{C} = \{\overline{a} : a \in C\}$ for any clause C. We now define a sequence $S_0, S_1, \ldots S_n$. Let $S_0 = \emptyset$, and for $i = 1$ to n define

$$S_i = \begin{cases} S_{i-1} \cup \{x_i\} & \text{if there exists } C \in r(\mathcal{F}) \text{ such that } \tilde{C} \subseteq S_{i-1} \cup \{\neg x_i\} \\ S_{i-1} \cup \{\neg x_i\} & \text{otherwise.} \end{cases}$$

We will show that S_n satisfies all the clauses in $\mathcal{F}$ and has no complementary literals.

By the construction, S_n cannot contain complementary literals, so it meets the first condition for being a partial satisfying assignment for $\mathcal{F}$. Now we shall prove that S_n meets the second condition by showing that for all $C \in r(\mathcal{F})$, $S_n \cap C \neq \emptyset$. It suffices to show that for all $C \in r(\mathcal{F})$, $\tilde{C} \not\subseteq S_n$. This is done in the following claim:

Claim: $\forall i \forall C \in r(\mathcal{F}), \tilde{C} \not\subseteq S_i$.

Proof of Claim: The proof is by induction on i:

Basis: $i = 0$. Let $C \in r(\mathcal{F})$. By assumption, $C \neq \emptyset$, so $\tilde{C} \neq \emptyset$, and hence $\tilde{C} \not\subseteq S_0 = \emptyset$.

Inductive step: Assume claim holds for $i - 1$. There are two cases to consider depending on whether or not x_i was chosen to be added to S_{i-1} to form S_i.

Case 1. $S_i = S_{i-1} \cup \{\neg x_i\}$. In this case we have for all $C \in r(\mathcal{F})$, $\tilde{C} \not\subseteq S_{i-1} \cup \{\neg x_i\} = S_i$. Thus we are done.

Case 2. $S_i = S_{i-1} \cup \{x_i\}$. In this case there is $C \in r(\mathcal{F})$ such that $\tilde{C} \subseteq S_{i-1} \cup \{\neg x_i\}$. By the inductive hypothesis $\tilde{C} \not\subseteq S_{i-1}$. Hence, $x_i \in C$.

For purposes of coming to a contradiction assume there is $D \in r(\mathcal{F})$ such that $\tilde{D} \subseteq S_i$. By the inductive hypothesis, $\tilde{D} \not\subseteq S_{i-1}$. Since $S_i = S_{i-1} \cup \{x_i\}$ then $\neg x_i \in D$. Thus, the set $E = (C - \{x_i\}) \cup (D - \{\neg x_i\})$ must also be in $r(\mathcal{F})$. However, both both $\tilde{C} - \{\neg x_i\}$ and $\tilde{D} - \{x_i\}$ are contained in S_{i-1}. We have $\tilde{E} = \tilde{C} - \{\neg x_i\} \cup \tilde{D} - \{x_i\}$. Hence, $\tilde{E} \subseteq S_{i-1}$, contradicting the inductive hypothesis. □