Discrete Mathematics (II) Spring 2012

Lecture 9: Deduction from Premises,Compactness, and Applications

Lecturer: Yi Li

1 Overview

In lecture 7, we introduce a tableau proof system to prove the validity of a proposition. But we should handle whether α is a logic consequence of Σ or not. Here, Σ is called a set of premises.

Mathematical Logic is a corner stone of modern computer science. In this lecture, we also show you how to apply propositional logic to other problems.

2 Deduction from Premises

Because a set of premises is introduced, we should define a new type of tableau proof system to handle the premises.

If α is a consequence of Σ, then any $\mathcal{V}$ satisfing Σ should also make α true. The problem can be simplified by considering the case when $\alpha_i \in \Sigma$. Then, we can introduce a proposition α_i into tableau in the form of $T\alpha_i$.

Definition 1 (Tableaux from premises)**.** *Let Σ be (possibly infinite) set of propositions. We define the finite tableaux with premises from Σ by induction:*

1. *Every atomic tableau is a finite tableau from Σ*
2. *If τ is a finite tableau from Σ and $\alpha \in \Sigma$, then the tableau formed by putting $T\alpha$ at the end of every noncontradictory path not containing it is also a finite tableau from Σ.*
3. *If τ is a finite tableau from Σ, P a path in τ, E an entry of τ occurring on P and $\prime\tau$ is obtained from τ by adjoining the unique atomic tableau with root entry E to the end of the path P, then $\prime\tau$ is also a finite tableau from Σ.*

If $\tau_0, \ldots, \tau_n, \ldots$ is a (finite or infinite) sequence of finite tableaux from Σ such that, for each $n \geq 0, \tau_{n+1}$ is constructed from τ_n by an application of (2) and (3), then $\tau = \cup\tau_n$ is a tableau from Σ.

Similarly, we can redefine tableau proof as following:

Definition 2. *A tableau proof of a proposition α from Σ is a tableau from Σ with root entry $F\alpha$ that is contradictory, that is, one in which every path is contradictory. If there is such a proof we say that α is provable from Σ and write it as $\Sigma \vdash \alpha$.*

To CST, we have the following property.

Theorem 3. *Every CST from a set of premises is finished.*

Unlike the CST in previous lecture, we can not guarantee the tableau to be finite. Because Σ could be a inifinite set. It means we can introduce premises as many as possible into our tableau.

Anyway, every entry in CST would be finally reduced. We can give several ways to expand CST in detail. Even if the proposition is an infinite sequence, we can also obtain the same property.

3 Soundness Theorem

We first give the following lemma.

Lemma 4. *If a valuation $\mathcal{V}$ makes every $\alpha \in \Sigma$ true and agrees with the root of a tableau τ from Σ, then there is a path in τ every entry of which agrees with $\mathcal{V}$.*

It sets up a connection between truth valuation $\mathcal{V}$ and sign of entries along noncontradictory path. When giving proof, we need handle propositions introduced from Σ whose corresponding entry is always with sign T.

By this lemma, we can easily prove the following soundness theorem.

Theorem 5 (Soundness)**.** *If there is a tableau proof of α from a set of premises Σ, then α is a consequence of Σ, i.e. $\Sigma \vdash \alpha \Rightarrow \Sigma \vDash \alpha$.*

Soundness theorem means that a propositon must be a logic consequence of Σ if it can be deduced from Σ.

4 Completeness Theorem

Similarly, we have the following Lemma.

Lemma 6. *Let P be a noncontradictory path in a finished tableau τ from Σ. Define a valuation $\mathcal{V}$ as the last section, then it agrees with all entries on P and so in particular makes every proposition $\beta \in \Sigma$ true.*

It is nearly the same as previous proof for completeness theorem without premises except that propositions from Σ should be handled specially.

With this lemma, we have completeness theorem.

Theorem 7 (Completeness)**.** *If α is consequence of a set Σ of premises, then there is a tableau deduction of α from Σ, i.e., $\Sigma \vDash \alpha \Rightarrow \Sigma \vdash \alpha$.*

Completeness theorem says that if a propostion is a consequence of a set of propositions, it can be deduced from this set.

5 Compactness Theorem

Theorem 8 (finite proof). *If $\tau = \cup\tau_n$ is a contradictory tableau from Σ, then for some m, τ_m is a finite contradictory tableau from Σ. In particular, if a CST from Σ is a proof, it is finite.*

It means a tableau proof is a finite sequence of trees.Otherwise we can imply contradiction that the tableau is not a tabeau proof.

With finite proof theorem, we have a type of compactness theorem by using completeness and soundness theorem.

Theorem 9. *α is a consequence of Σ if and only if α is a consequence of some finite subset of Σ.*

Sketch. We apply completeness theorem first and get $\Sigma \vdash \alpha$. For any tableau proof must be finite, we can collect all proposition introduced from Σ and form a finite set Σ_0. The tableau also shows $\Sigma_0 \vdash \alpha$. We then apply soundness theorem and obtain $\Sigma_0 \models \alpha$. It is proved now. □

However, compactness theorem can be directly proved. We have the following version.

Theorem 10 (Compactness). *Let $\Sigma = \{\alpha_i | i \in \omega\}$ be an infinite set of a propositions. Σ is satisfiable if and only if every finite subset Γ of Σ is satisfiable.*

Generally, if $\mathcal{V}_1$ and $\mathcal{V}_2$ satisfy Σ_1 and Σ_2 respectively, it does not mean the union of two truth valuation would satisfy all propositions in two sets.

Sketch. Consider the tableau with root

$$F(\neg\alpha \wedge \alpha).$$

We already know that $(\neg\alpha \wedge \alpha)$ is always false, which means that any truth valuation $\mathcal{V}$ always agrees with the root entry. If it is a finite tableau, every path of tableau is contradictory. It means that all propositions in Σ along a contradictory path is unsatisfiable. It is contradictory to every finite subset is satisfiable. So there is a infinite path in the tableau. □

This compactness theorem transform a problem with infinite propositions to a infinite sequence of finite propositions.

6 Circuit and Proposition

Let's take $0, 1$ as F, T respectively. We can use circuit gate to represent $\wedge, \vee$ and $\neg$ as *and,or,not* respectively. So given any proposition, we can design a circuit to compute the truth value with the specific input.

Example 1. *Consider the circuit*[1] *for the following propositions:*

[1]The circuit will be drawn in the future.

1. $(A_1 \wedge A_2) \vee (\neg A_3))$

2. $(A \wedge B \wedge D) \vee (A \wedge B \wedge \neg C)$

The complexity of circuits depends on the complexity of proposition. We usually call the depth of proposition as *delay* of circuit and the number of gates as *power consumption*. To design a good circuit, we try to minimize delay and power consumption.

Example 2. *Consider the boolean function majority of* $\{A, B, C\}$. *It means that the value of function depends on the majority of input.*

Solution: We first consider its truth value table, we can simple connectives to represent majority.

$$\begin{aligned} m(A, B, C) &= (A \wedge B \wedge C) \vee (A \wedge B \wedge \neg C) \vee (A \wedge \neg B \wedge C) \vee (\neg A \wedge B \wedge C) \\ &= (B \wedge C) \vee (A \wedge C) \vee (A \wedge B) \\ &= (A \wedge (B \vee C)) \vee (B \wedge C) \end{aligned}$$

In another way, we can depict the boolean function with a state diagram and then design the circuit accroding it. This method will be introduced in the successor course *Digital Component Desing.*

7 Formalize Problems with Propositions

Example 3. *Suppose there is a murder case with three suspects. The police queried them about murder case.*

A said,"I didn't do it. The victim is a friend of B. And C hates him." B said, "I didn't do it. Even I don't know him. And I am not present." C said, "I didn't do it. I saw A and B stayed with the victim in that day. The murder must be one of them."

Solution: Suppose only murder would lie. We try to formalize it with the following propositions:

1. A: A killed victim.
2. BKV: B knows the victim.
3. AP: A is present.
4. CHV: C hates the victim.
5. $(A \wedge \neg B) \vee (\neg A \wedge B)$: murder is either A or B.

Now we can represent the satement of each sucpects as following:

1. A: $\neg A \wedge BKV \wedge CHV$.
2. B: $\neg B \wedge \neg BKV \wedge \neg BP$.
3. C: $\neg C \wedge AP \wedge BP \wedge ((A \wedge \neg B) \vee (\neg A \wedge B))$.

It is easy to know that the maximal satisfiable set of propositions can only contain A's and C's statement. Then we can imply that B would be the murder.

Example 4. *Consider the pigeonhole principle: $f : n^+ \to n, \exists i, j, f(i) = f(j)$, where $0 \le i < j \le n$.*

Solution: let p_{ij} means $f(i) = j$. Then we can describe everywhere defined property as

$$\alpha_1 = \wedge_{0 \le i \le n} \vee_{0 \le j < n} p_{ij}$$

and we can describe single value as

$$\alpha_1 = \wedge_{0 \le i \le n} \wedge_{0 \le j \ne k < n} \neg(p_{ij} \wedge p_{ik})$$

Now we can describe pigeonhole principle as

$$\varphi = (\alpha_1 \wedge \alpha_2) \wedge (\vee_{0 \le i < j \le n} \vee_{0 \le k < n} (p_{ik} \wedge p_{j,k}))$$

Remark: This form of pigeonhole is much harder to recognize than its original version. However, it is described by a much more rigid language compared with our natural language, which could result in ambiguities.

In fact, mathematical logic can be applied into program verification, model checking, and such other applications. European Space Agency only uses software which has gotten through program verification.

8 Application of Compactness Theorem

Compactness theorem is a very important result in mathematical logic. It establishes a connection between infinity and finitude.

Example 5. *Given an infinite planar graph. If its every finite subgraph is k-colorable, then the graph itself is also k-colorable.*

A graph is $G =< V, E >$, where V is the set of vertices and $E \subseteq V^2$ is the set of all edges. If $(a, b) \in E$, it is denoted as aEb sometimes. G is k-colorable if V can be decomposed into k-color classes $V = C_1 \cup C_2 \cup \cdots \cup C_k$, where $C_i \ne \emptyset$ and $C_i \cap C_j = \emptyset$ if $i \ne j$. It is obvious $(a, b) \notin E$ if $a, b \in C_i$.

A finite subgraph is k-colorable. It means there is a k-colorable graph $G_0 =< V_0, E_0 >$, where $V_0 \subset V$ is a finite set and $E_0 \subset E$ is the set determined by V_0.

Proof. Let $p_{a,i}$ represent vertex a is colored with i. We can formulate a graph which is k-colorable with the following propositions.

1. $p_{a,1} \vee p_{a,2} \vee \cdots \vee p_{a,k}$, for every $a \in V$. It means every vertex could be colored with at least one of k colors.

2. $\neg(p_{a,i} \wedge p_{a,j}), 1 \le i < j \le k$ for all $a \in V$. It means $C_i \cap C_j = \emptyset$.

3. $\neg(p_{a,i} \wedge p_{b,i}), i = 1, \ldots, k$ for all aEb. It means no neighbors have the same color .

Then we get a set X with infinite propositions. For any finite subset $X_0 \subset X$, we can extract vertices V_0 from it and construct a set X_1 which describe the graph G_0 generated by V_0. For every finite subgraph is k-colorable, X_1 is satisfiable. X_0 must be satisfiable.

According to compactness theorem, X is satisfiable which means the graph is k-colorable. □

Remark: Generally, this theorem is not easy to prove via graph approach. Because there is no effective way to find it possible that a big graph is still k-colorable merged by two k-colorable graph. However, Compactness Theorem does not need this requirement. Here, we should be aware that a set of propositions is constructed and we try to prove that every finite subset of it is satisfiable, which is the essence of Compactness Theorem.

Example 6. *Every set S can be (totally) ordered.*

Similarly, it can be proved like k-colorable infinite graph. The point is to represent our problem with a set of propositions. This is left as an exercise.

Hints: A set is partial order at least. If you can change a partially ordered set into a linear order set, you successfully complete the proof.

9 König Lemma and Compactness Theorem

In our textbook, Compactness Theorem is proved by König lemma. Now we will show you that it can be proved by Compactness Theorem.

Lemma 11 (König). *A infinite tree with finite branch has a infinite path.*

Actually, the problem can be represented as following. If every $a \in T$ has only finitely many immediate successor and T contains arbitrarily long finite paths, then there is a infinite path in T starts at root.

Proof. Tree is a hierarchical structure. It means that the vertices of a tree could be divided into many sets which corresponds to vertices in some level. So we can define $S_0 = \{c|c \text{ is the root}\}$ and $S_k = \{b \in T| \text{ there is a } a \in S_k \text{ and } b \text{ is a immediate successor of } a\}$. For every k, S_k is finite and no S_k is empty because of infinity of given tree.

Denote p_a as that vertex a is in path P. We now represent a tree with a set of propositions, Σ, as following:

1. $\vee_{a \in S_k} p_a$: there is at least one vertex of level k in a path;
2. $\vee_{a,b \in S_k} \neg(p_a \wedge p_b)$: there is only one vertex of level k in path p.
3. $p_a \to p_b$: b is a immediate successor of a.

For there are infinite vertices, we have infinite propositions. If they are all satisfiable. We do know there is a infinite path. Given a subset Σ_0, it is just a part of subtree with height k. Then we just add missed propositions into Σ_0 and obtain the subtree represented by Σ_0'. As the tree has arbitrarily long finite paths. We know Σ_0' is satisfiable and also Σ_0. Now we just apply compactness theorem to get final result. □

In textbook, compactness theorem of proposition logic is proved based on König lemma. Here we prove inversely.

Exercises

1. Let Σ be finite set of propositions and $\wedge\Sigma$ the conjunction of its members. Prove that for any proposition α the following are equivalent:
   (a) $\Sigma \models \alpha$.
   (b) $\models \wedge\Sigma \to \alpha$.
   (c) $\Sigma \vdash \alpha$
   (d) $\vdash \wedge\Sigma \to \alpha$.
2. Suppose Σ is a finite set of propositions. Show that every CST from Σ is finite.
3. Prove $\{A \vee \neg B, B \vee \neg C, C \vee \neg D\} \models (D \to A)$.
4. Suppose proposition could be infinite long. Define a new method to construct a CST and prove that every CST is finished.
5. Design a circuit for multiply with two two bits input and four bits output. For example, we have $1 * 1 = 1, 10 * 11 = 110$.