Discrete Mathematics (II) Spring 2020

Lecture 5: Formation Tree, Truth Assignments, and Valuations

Lecturer: Yi Li

1 Overview

In this lecture, we first completely focus on syntax of proposition. And the term “proposition” does always mean well-defined proposition without explicit declaration.

The most important topic is to make sure the unique readability of a well defined proposition. And then a proposition is associated with a tree, called formation tree. Therefore, we find the properties of a well-defined proposition based on its tree form. Finally, we introduce a parsing algorithm to check the validity of a sequence of symbols, with a formation tree as a by-product.

Then we define truth assignments and valuations in order to make truth table not very important, which is tedious. Finally, a truth valuation can be determined uniquely by a truth assignment. Sometimes, we call it the semantics of propositional logic. Correspondingly, well-defined proposition is called the syntax of propositional logic.

Given a set of propositions and a proposition, we can bind them in the point of view of truth valuation. Here we only connect them by truth valuation but syntax.

2 Some properties of proposition

2.1 Ambiguity of a sentence

In everyday language, we sometimes could speak some sentence which could be understood totally with different meaning, which is called ambiguous.

Example 1. *Consider the following sentences:*

1. *The lady hit the man with an umbrella.*
2. *He gave her cat food.*
3. *They are looking for teachers of French, German and Japanese.*

Once you read these sentences, you way wonder:

1. Is the lady using an umbrella to hit or is she hitting a man who is carrying an umbrella?
2. Is he giving cat food to her or is he giving her cat some food?
3. Are they looking for teachers who can each teach one language or all three languages?

You may find more examples. Here, the ambiguity is not caused by the meaning/semantics of some word. The reason is just the way to group words in different approaches.

Similarly, you could encounter the same trouble when you write a piece of segment of code. The example is the way to nest if and else. You can also consider the following expression "$4/2 \times 2$".

Without the help of parentheses, proposition could also run into the same trouble. Consider the following example.

Example 2. *Consider the following proposition*

$$A_1 \vee A_2 \wedge A_3,$$

which is definitely not well-defined.

Solution. We have two possible different propositions

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

Of course, they have different abbreviation truth tables. □

Mathematics is rigid. One of the responsibilities of mathematical logic is to eliminate ambiguity from mathematics. So we should first eliminate ambiguity in mathematical logic.

2.2 Parentheses

Finally, we want to design a systematic approach to recognize/parse a well defined proposition. But we first dig some properties of well defined propositions.

A proposition is a sequence of symbols generated by proposition language. We just take it as a string. Parentheses are very important to construct a well-defined proposition. We just count the number of left and right parenthesis and have the following theorem.

Theorem 1. *Every well-defined proposition has the same number of left as right parentheses.*

Proof. Consider the symbols without parenthesis first. They are the simplest cases: propositional letters. And we know that a complicated proposition consists of two or one simpler proposition(s) with a connective. If a simpler one keep the property, we can derive the invariant for the complicated one.

Obviously, we can prove it by induction to investigate all propositions by following the approach to construct a compound proposition. □

What metric is chosen for induction? It is magic. Here, we actually use the depth of nested connectives, which will be introduced in the next section. Some guy may take the number of parenthesis. It is difficult to characterize the feature and to apply inductive proof. The exercise will demonstrate this.

Theorem 2. *Any proper initial segment of a well defined proposition contains an excess of left parenthesis. Thus no proper initial segment of a well defined proposition can itself be a well defined propositions.*

Go back to lecture three if your forget what is a proper initial segment.

Proof. Prove it by induction from simple to complicated propositions. And we also need Theorem 1. □

3 Formation Tree

A proposition is a sequence of symbols. The structure determines the reading of a proposition. In this section, we are to map a proposition to a tree, named formation tree. It will help us to read a proposition.

Definition 3 (Top-down). *A formation tree is a finite tree T of binary sequences whose nodes are all labeled with propositions. The labeling satisfies the following conditions:*

1. *The leaves are labeled with propositional letters.*
2. *if a node σ is labeled with a proposition of the form $(\alpha \vee \beta), (\alpha \wedge \beta), (\alpha \rightarrow \beta)$ or $(\alpha \leftrightarrow \beta)$, its immediate successors, $\sigma\hat{}0$ and $\sigma\hat{}1$, are labeled with α and β (in that order).*
3. *if a node σ is labeled with a proposition of the form $(\neg \alpha)$, its unique immediate successor, $\sigma\hat{}0$, is labeled with α.*

How to map a well-defined proposition to a tree? Consider the following example.

Example 3. *The formation tree of $(A \vee B), ((A \wedge B) \rightarrow C)$.*

It is easy to draw its formation tree for this example. However, when a proposition is complicated enough, it is not easy. Just rethink your approach. We may follow the procedure of constructing a proposition from a propositional letters in a bottom-up way, even we define it in a top-down approach.

These two examples are specified. What about the other propositions? Given proposition, we have the following questions:

1. Is there a formation tree?
2. Is it unique?

It is lucky that we have the following Theorem.

Theorem 4. *Each well-defined proposition has a unique formation tree associated with it.*

Proof. (Sketch) It is divided into two parts. Both are inductive proof on depth of a proposition.

1. Existence of the formation tree by induction on depth.
2. Uniqueness of the formation tree by induction on depth.

□

From now on, we sometimes call the labeled tree of a proposition associated formation tree.

Once we have formation tree of a proposition, we can define these following terms.

Definition 5. *Given an associated formation tree, we have*

1. *The depth of a proposition is the depth of associated formation tree.*
2. *The support of a proposition is the set of propositional letters that occur as labels of the leaves of the associated formation tree.*

Actually, all inductive proof could be based on the depth of a proposition.

4 Assignments and Valuations

Propositional letters are the simplest propositions. There is no constraint between each other. We just define an operation, called assignment, which assigns a value *true* or *false* on every propositional letter.

Definition 6 (Assignment). *A truth assignment $\mathcal{A}$ is a function that assigns to each **propositional letter** A a unique truth value $\mathcal{A}(A) \in \{T, F\}$.*

Generally, a proposition is a sequence of symbols constructed according to some rules determined in previous lecture. Whether it is true or false can not be simply assigned like propositional letters.

Consider an example in Figure 1.

Example 4. *Truth assignment of α and β and valuation of $(\alpha \vee \beta)$.*

α	β	$(\alpha \vee \beta)$
T	T	T
T	F	T
F	T	T
F	F	F

Figure 1: Truth assignment and valuation

We can infer from the example that truth valuation of a proposition is determined by those propositions which it is based on.

We define the following term to guarantee the truth of a compound proposition.

Definition 7 (Valuation). *A truth valuation $\mathcal{V}$ is a function that assigns to each **proposition** α a unique truth value $\mathcal{V}(\alpha)$ so that its value on a compound proposition is determined in accordance with the appropriate truth tables.*

Here, we should remember that truth valuation determines all propositions generated according to definition of well-defined proposition. Especially, when α is a propositional letter we have $\mathcal{V}(\alpha) = \mathcal{A}(\alpha)$ for some $\mathcal{A}$.

In Definition 7, it says that the value on a compound proposition is determined in accordance with the appropriate truth tables. It means that we have $\mathcal{V}(\neg\alpha) = \neg\mathcal{V}(\alpha), \mathcal{V}(\alpha \vee \beta) = \mathcal{V}(\alpha) \vee \mathcal{V}(\beta)$ and the others.

Generally, we have the following theorem:

Theorem 8. *Given a truth assignment $\mathcal{A}$ there is a unique truth valuation $\mathcal{V}$ such that $\mathcal{V}(\alpha) = \mathcal{A}(\alpha)$ for every propositional letter α.*

Proof. The proof can be divided into two step.

1. Construct a $\mathcal{V}$ from $\mathcal{A}$ by induction on the depth of the associated formation tree.

2. Prove the uniqueness of $\mathcal{V}$ with the same $\mathcal{A}$ by induction bottom-up.

□

It shows us the relation between truth assignment and truth valuation. Actually, truth assignment and valuation characterize the semantics of proposition logic from different views. For simplicity of theory, one is enough. For convenience, both are needed to make statement simple.

In the Definition on truth assignment and valuation, all proposition letters and propositions respectively are considered. We have known that there are infinitely many objects. It is unnecessary for us in computer science for computer can only handle finite objects. In practice, the number of propositions we should consider is also finite. And each proposition has also finite length. Therefore, we sometimes just consider a specific proposition α. Then there are only finite propositional letters taken into consideration. There is a corollary.

Corollary 9. *If $\mathcal{V}_1$ and $\mathcal{V}_2$ are two valuations that agree on the support of α, the finite set of propositional letters used in the construction of the proposition of the proposition α, then $\mathcal{V}_1(\alpha) = \mathcal{V}_2(\alpha)$.*

Two valuations as a whole are different indeed. In actually we only handle cases with finite number of proposition letters. Then given a proposition when the assignments determined agree with each other on support. They have the same truth value on a proposition.

Given a proposition, there is a case that it is always true whatever the truth valuation is.

Definition 10. *A proposition σ of propositional logic is said to be valid if for any valuation $\mathcal{V}, \mathcal{V}(\sigma) = T$. Such a proposition is also called a tautology.*

Example 5. *$\alpha \vee \neg\alpha$ is a tautology.*

Solution:

α	$\neg\alpha$	$\alpha \vee \neg\alpha$
T	F	T
F	T	T

□

This tautology can not convey useful information. Because we just talk about both right and wrong side together. To this case, we do think tautology nonsense. But it represents a very special set of propositions independent of truth valuation, say a structure with different equivalent views. The Adequacy Theorem implies that every proposition has at least one equivalent form either DNF or CNF.

Syntax of proposition logic make sure that two string are the same proposition if they are the same symbol sequence. Semantics will bring us more profound result. Consider the following example.

α	β	$\alpha \to \beta$
T	T	T
T	F	F
F	T	T
F	F	T

α	β	$\neg\alpha$	$\neg\alpha \vee \beta$
T	T	F	T
T	F	F	F
F	T	T	T
F	F	T	T

Figure 2: Logically equivalent propositions

Example 6. $\alpha \to \beta \equiv \neg\alpha \vee \beta$.

Proof. Prove by truth table in Figure 2. □

Although, they have different formation tree. But they are the same if they are only characterized by truth valuation.

Definition 11. *Two proposition α and β such that, for every valuation $\mathcal{V}, \mathcal{V}(\alpha) = \mathcal{V}(\beta)$ are called logically equivalent. We denote this by $\alpha \equiv \beta$.*

With is definition, we can construct tautologies as many as possible. For $\alpha \equiv \beta$ can be represented as a proposition $\alpha \leftrightarrow \beta$.

5 Consequence

In practice, we often mention a pattern that a result can be inferred from some facts. We now consider this pattern from the point of view of semantics.

Definition 12. *Let Σ be a (possibly infinite) set of propositions. We say that σ is a consequence of Σ (and write as $\Sigma \models \sigma$) if, for any valuation $\mathcal{V}$,*

$$(\mathcal{V}(\tau) = T \text{ for all } \tau \in \Sigma) \Rightarrow \mathcal{V}(\sigma) = T.$$

Especially when $\Sigma = \emptyset$, its consequences are tautologies. For σ must be satisfied by every truth valuation.

Another extreme case is that no truth valuation can satisfy all propositions in Σ, which is also called *unsatisfied* defined later. Then every proposition is its consequence. Sometimes we call it vacuum/null satisfaction. It is mentioned for completeness for it can't give us some positive result.

Example 7. *Consider the following examples:*

1. *Let $\Sigma = \{\neg A \vee B\}$, we have $\Sigma \not\models B$.*

2. Let $\Sigma = \{A, \neg A \vee B\}$, we have $\Sigma \models B$.

3. Let $\Sigma = \{A, \neg A \vee B, C\}$, we have $\Sigma \models B$.

Definition 13. *We say that a valuation $\mathcal{V}$ is a model of Σ if $\mathcal{V}(\sigma) = T$ for every $\sigma \in \Sigma$. We denote by $\mathcal{M}(\Sigma)$ the set of all models of Σ.*

Example 8. *Let $\Sigma = \{A, \neg A \vee B\}$, we have following models:*

1. Let $\mathcal{A}(A) = T, \mathcal{A}(B) = T$
2. Let $\mathcal{A}(A) = T, \mathcal{A}(B) = T, \mathcal{A}(C) = T$.
3. Let $\mathcal{A}(A) = T, \mathcal{A}(B) = T, \mathcal{A}(C) = F, \mathcal{A}(D) = F, \ldots$.

Here just lists three of all models. It shows us that there are models as many as you wish once you can introduce new propositional letters. Actually the satisfaction depends only on its support set. So we just apply Corollary 9 in practice.

Definition 14. *We say that propositions Σ is* satisfiable *if it has some model. Otherwise it is called* invalid*.*

Reviewing example 7, we find that more consequence can be derived when Σ has more propositions. Generally, we have the following properties.

Proposition 15. *Let $\Sigma, \Sigma_1, \Sigma_2$ be sets of propositions. Let $Cn(\Sigma)$ denote the set of consequence of Σ and $Taut$ the set of tautologies.*

1. $\Sigma_1 \subseteq \Sigma_2 \Rightarrow Cn(\Sigma_1) \subseteq Cn(\Sigma_2)$.

2. $\Sigma \subseteq Cn(\Sigma)$.

3. $Taut \subseteq Cn(\Sigma) = Cn(Cn(\Sigma))$.

4. $\Sigma_1 \subseteq \Sigma_2 \Rightarrow \mathcal{M}(\Sigma_2) \subseteq \mathcal{M}(\Sigma_1)$.

5. $Cn(\Sigma) = \{\sigma | \mathcal{V}(\sigma) = T$ for all $\mathcal{V} \in \mathcal{M}(\Sigma)\}$.

6. $\sigma \in Cn(\{\sigma_1, \ldots, \sigma_n\}) \Leftrightarrow \sigma_1 \to (\sigma_2 \ldots \to (\sigma_n \to \sigma) \ldots) \in Taut$.

Proof. Proof of all except the property 6 just follows the definition of consequence. And you also need apply the techniques proving two sets which are equal. □

Theorem 16. *For any propositions φ, ψ, $\Sigma \cup \{\psi\} \models \varphi \Leftrightarrow \Sigma \models \psi \to \varphi$ holds.*

Proof. Prove by the definition of consequence.

When we consider $\Rightarrow$, $\mathcal{V}$ which satisfy Σ are divided into two parts, $\mathcal{V}_1(\psi) = T$ and $\mathcal{V}_2(\psi) = F$. Then we investigate whether $\mathcal{V}$ satisfies $\psi \to \varphi$.

Conversely, $\mathcal{V}$ which makes ψ false are discarded. Because they are not taken into consideration to satisfy $\Sigma \cup \{\psi\}$. □

With this Theorem 16, we can prove result 6 in Proposition 15 by induction.

Exercises

1. Show that there are no well-defined propositions of length $2, 3$, or 6, but that any other positive length is possible.

2. Given an example such that $(\alpha \wedge \beta) = (\gamma \wedge \delta)$ but $\alpha \neq \gamma$, where α and β are well-defined and γ and δ are two expressions.

3. Draw formation tree of the following propositions:
   (a) $((A \vee B) \to ((\neg C) \wedge D))$.
   (b) $(((\neg A) \vee B) \wedge (A \vee (\neg B)))$.

4. Let α be a well-defined proposition; let c be the number of places at which binary connective symbols($\wedge, \vee, \to, \leftrightarrow$) occur in α; let s be the number of places at which proposition letters occur in α. Show by using the induction principle that $s = c + 1$.

5. Design an algorithm to determine whether a given string is a well-defined proposition or not.

6. Check whether the following propositions are valid or not
   (a) $(A \to B) \leftrightarrow ((\neg B) \to (\neg A))$
   (b) $A \wedge (B \vee C) \leftrightarrow (A \wedge B) \vee (A \wedge C)$

7. Prove or refute each of the following assertions:
   (a) If either $\Sigma \models \alpha$ or $\Sigma \models \beta$, then $\Sigma \models (\alpha \vee \beta)$.
   (b) If $\Sigma \models (\alpha \wedge \beta)$, then both $\Sigma \models \alpha$ and $\Sigma \models \beta$.

8. Prove the following assertion:
   (a) $Cn(\Sigma) = Cn(Cn(\Sigma))$.
   (b) $\Sigma_1 \subset \Sigma_2 \Rightarrow \mathcal{M}(\Sigma_2) \subset \mathcal{M}(\Sigma_1)$.
   (c) $Cn(\Sigma) = \{\sigma \mid \mathcal{V}(\sigma) = T \text{ for all } \mathcal{V} \in \mathcal{M}(\Sigma)\}$.

(d) $\sigma \in Cn(\{\sigma_1, \ldots, \sigma_n\}) \Leftrightarrow \sigma_1 \to (\sigma_2 \ldots \to (\sigma_n \to \sigma) \ldots) \in Taut$.

9. Suppose we have two assertions, where α and β both are propositions and Σ is a set of propositions:

   (a) If $\Sigma \models A$, then $\Sigma \models B$.

   (b) $\Sigma \models (A \to B)$.

   Show the relation between them. It means whether one can imply another.