# Handout 7

# Predicate Logic Trees

In a previous lesson (Lesson 4), we saw that a truth-tree method could be developed for arguments in $PL$ in order to mechanically determined whether a $PL$-argument was deductively valid or invalid. We also saw that $PL$ is unable to express all of the arguments that we intuitively regard as valid and so we developed a new, more expressive language to express these arguments. This new language is the language of predicate logic or $RL$. In this lesson, we adapt the truth-tree method for $RL$. Unfortunately, there is no complete decision procedure for $RL$. That is, while we can use the truth-tree method to mechanically determine whether a number of arguments are valid, this method will fail to be a completely mechanical process that will always give us a "yes" or "no" answer to the question of whether or not the argument is valid or invalid. In other words, $RL$ is semi-decideable.

### 7.0.2 Predicate trees: Decomposition Rules

In addition to the truth-tree decomposition rules from propositional logic, there are four additional decomposition rules for predicate logic.

$$\neg(\exists x)\mathbf{P}\checkmark$$
$$(\forall x)\neg(\mathbf{P})$$

**Figure 7.1** – Negated Existential Decomposition ($\neg\exists D$)

$$\neg(\forall x)\mathbf{P}\checkmark$$
$$(\exists x)\neg(\mathbf{P})$$

**Figure 7.2** – Negated Universal Decomposition ($\neg\forall D$)

$$(\exists x)\mathbf{P}\checkmark$$
$$\mathbf{P}(a/x)$$

**Figure 7.3** – Existential Decomposition ($\exists D$), where $a$ is an individual constant (name) that does not previously occur in the branch

$$(\forall x)\mathbf{P}$$
$$\mathbf{P}(a/x)$$

**Figure 7.4** – Universal Decomposition ($\forall D$), where $a$ is any individual constant (name)

**Example 1:**

| | | |
|---|---|---|
| 1 | $\neg(\forall x)Px\checkmark$ | $P$ |
| 2 | $\neg(\exists y)Ryy\checkmark$ | $P$ |
| 3 | $(\exists x)\neg Px$ | $1\neg\forall D$ |
| 4 | $(\forall y)\neg Ryy$ | $2\neg\exists D$ |

**Example 2:**

| | | |
|---|---|---|
| 1 | $\neg(\forall x)Px\checkmark$ | $P$ |
| 2 | $\neg(\exists y)Ryy\checkmark$ | $P$ |
| 3 | $(\exists x)\neg Px\checkmark$ | $1\neg\forall D$ |
| 4 | $(\forall y)\neg Ryy\checkmark$ | $2\neg\exists D$ |
| 5 | $\neg Pa$ | $3\exists D$ |
| 6 | $\neg Raa$ | $4\forall D$ |
| 7 | $\neg Rbb$ | $4\forall D$ |

**Example 3:**

| | | |
|---|---|---|
| 1 | $(\exists x)Px$ | $P$ |
| 2 | $(\exists x)Qx$ | $P$ |
| 3 | $Pa$ | $1\exists D$ |
| 4 | $Qa$ | $2\exists D$, *NO*! |

### 7.0.3 Strategies & Terminology

In $PL$, a completed open branch is defined as a fully decomposed branch that is not closed. For trees in $RL$, a new definition is required since branches involving

universally quantified propositions cannot be fully decomposed when a domain has an infinite number of items.

DEFINITION – COMPLETED OPEN BRANCH

A branch is a completed open branch if and only if (1) all complex propositions that can be fully decomposed are decomposed, (2) for all universally quantified propositions $(\forall x)\mathbf{P}$ occurring in the branch, there is a substitution instance $\mathbf{P}(a/x)$ for each constant in that branch, and (3) the branch is not a closed branch.

**Example 1:**

| | | |
|---|---|---|
| 1 | $Pa$ | $P$ |
| 2 | $Rb$ | $P$ |
| 3 | $Lcc$ | $P$ |
| 4 | $(\forall x)Px$ | $P$ |
| 5 | $Pa$ | $4\forall D$ |
| 6 | $Pb$ | $4\forall D$ |
| 7 | $Pc$ | $4\forall D$ |

At line 4, it is important to note clause (2) of the definition of a completed open branch. Clause (2) says that "for all universally quantified propositions $(\forall x)P$ occurring in the branch, there is a substitution instance $P(a/x)$ for each constant in that branch." Note that there are three names in the above branch containing $(\forall x)Px$ ($a$, $b$, and $c$), thus, $(\forall x)Px$ should be decomposed (using $\forall D$) using each of these as substitution instances.

**Example 2:**

| | | |
|---|---|---|
| 1 | $(\exists x)Px$ | $P$ |
| 2 | $(\forall x)\neg Px$ | $P$ |
| 3 | $\neg Pa$ | $2\forall D$ |
| 4 | $Pb$ | $1\exists D$ |
| | *NO*! | |

In the above example, the universally quantified proposition at line (2) is not decomposed for each name in the branch. For this reason, this branch cannot be claimed as open.

## 7.1 Strategic Rules for Decomposing Predicate Truth-Trees

The strategic rules for $RL$ are an extension of the strategic rules used for $PL$ trees.

1. Use no more rules than needed.
2. Decompose negated quantified expressions and existentially quantified expressions first.
3. Use rules that close branches.
4. Use stacking rules before branching rules.
5. When decomposing universally quantified propositions, it is a good idea to use constants that already occur in the branch.
6. Decompose more complex propositions before simpler propositions.

**Exercise 1:** *Using the new strategic rules as your guide, decompose the following formulas:*

1. $(\forall x)(Px \rightarrow Rx), Pa, \neg Rb$
2. $(\exists x)\neg(Px \vee Rx), (\exists y)\neg(Py \vee Ry), (\forall z)(Pz \vee Rz)$
3. $\neg(\forall x)Px \rightarrow (\exists z)Pz, \neg(\exists x)\neg Px$
4. $(\forall x)\neg Mx, (\forall x)Px, Pa \rightarrow (\exists x)Mx$
5. $Laa, (\exists x)(\exists y)Zxy, (\forall x)(\forall y)\neg Zxy$
6. $(\exists x)(\exists y)(Pxy \wedge Lxy), \neg(\exists x)(\exists y)Pxy \vee \neg(\exists x)(\exists y)Lxy$

## 7.2 Predicate Truth Trees: Analysis

Truth trees can be used to determine various semantic properties about propositions, sets of propositions, and arguments. For example, you can use a truth tree to mechanically determine whether or not an argument is valid or invalid.

Knowing that whether or not an argument is valid or invalid is helpful, but truth trees can do more. If the argument is invalid, a tree can be used to recover a *model* that provides an example of that argument's invalidity. That is, we can use a truth tree to create a scenario that shows one case where the argument we are considering is invalid.

This short lesson teaches you how to recover a model from a truth tree that indicates its argument is invalid.

### 7.2.1 What is a Model: In Simple Terms

What is a model? In this section, I provide a definition of a model and try to explain this definition in simpler terms. In the next section, I give an explanation of how to recover a model from a truth tree with a completed open branch.

DEFINITION – MODEL

A **RL**-model ($\mathcal{M}$) is an ordered pair $M = \langle \mathcal{D}, \mathscr{I} \rangle$ where

- $\mathcal{D}$ is a non-empty set
- $\mathscr{I}$ is an interpretation function that
  1. assigns sets of n-tuples of objects of $\mathcal{D}$ to n-place predicates
  2. assigns objects of $\mathcal{D}$ to names

There is some jargon in this definition that you may have forgotten or not be clear on. Let's explain each part of this definition in simple terms.

A model is a two-part structure. It is a pair of things like two people in a relationship or two people holding hands or two teammates.

The first part of the model is the *domain of discourse* or the *domain*. This part of the model is abbreviated as $\mathcal{D}$ and it refers to all of the things we want to talk about. We can think of the domain in even simpler terms. It consists of all of the objects under discussion. It includes the fluffy cloud we see in the sky, individual stars on a starry night, the people in a room, the natural numbers, etc. We often indicate what items are in the domain, e.g. movies, books, laws, people, either by listing all of them one by one or by indicating a property that they all share.

**Example 1: Two Ways to Illustrate a Domain**

1. $\mathcal{D} = \{a, b, c, d\}$
2. $\mathcal{D} = \{x \mid movies\}$
3. $\mathcal{D} = \{x \mid$ days of the week $\}$

The other part of the model is the *interpretation of* **RL** or the *interpretation*. This part of the model is abbreviated as $\mathscr{I}$ and it refers to two things. First, for each name in the formal language **RL**, it refers to the corresponding object in the domain. More simply, an interpretation gives each name $(a, b, c, d)$ a corresponding referent or object in the domain. Just as proper names in English, e.g. "George Washington", refer to people in our world, names in **RL** refer to objects in the domain.

We abbreviate the interpretation of names as follows:

- $\mathscr{I}(a) = a$, this says the name "a" is interpreted as the object $a$ in the domain $\mathcal{D}$
- $\mathscr{I}(Mark) = Mark$, this says the name "Mark" is interpreted as the object $Mark$ in the domain $\mathcal{D}$
- $\mathscr{I}(b) = b$, this says the name "a" is interpreted as the object $b$ in the domain $\mathcal{D}$

It is important to note that what is being interpreted is a *name* in terms of an item in the domain. Thus, an interpretation of the name "George Washington" is given in terms of the person *George Washington*. That is, $\mathscr{I}$(George Washington) = *George Washington*. Thus, $\mathscr{I}(a) = a$ says that the name "a" is interpreted in terms of an object $a$ in the domain.

> Interpreting names in terms of objects in the domain often looks trivial: "a" refers to $a$ and "George Washington" refers to *George Washington*. In each case, it appears that for each name, we always know the object in the domain to which it refers. This, however, is not always the case as discovering the referent of a name is often a process of discovery.

1. Consider a domain of discourse that consists of all of the people on this planet. Now consider a series of unsolved murders that occurred in the 1990s along interstate 70 (an interstate highway that runs from Maryland to Utah). This murderer is known as the "Interstate 70 Killer". We will abbreviate "Interstate 70 Killer" as $k$. In this example, $k$ refers to an item in the domain but we don't know which object.

2. Every object is identical to itself. That is, it seems trivially true that $a = a$. We don't need any experience to know this. If a name $a$ is interpreted in terms of an object in the domain, and we know what object it refers to, then it might seem that if $b$ has the same interpretation as $a$ (that is, it refers to the same object), then it is trivially true that $a = b$. And, it might appear that we wouldn't need any special experience to know this. This, however, is not the case.

   Suppose you are ready to go to sleep. Before you do, however, you go outside and see what you take to be a star in the sky. You name that object "Hesperus". We will abbreviate "Hesperus" as $h$. When you wake up in the morning, you go outside and see what you take to be a star. You name that object "Phosphorus". We will abbreviate "Phosphorus" as $p$. Now you have two names for two different phenomena. Hesperus refers to the evening star while Phosphorus refers to the morning star. What you don't know, however, is that "Hesperus" and "Phosphorus" refer to the same object, an object that is not a star at all but the planet Venus. That is, $\mathscr{I}(h) = \mathscr{I}(p)$.

Second, an interpretation also assigns each n-place predicate in **RL** a set of objects in the domain. More simply, if we are thinking about the 1-place predicate "is red" or $R$, the interpretation simply assigns that predicate all of the red objects in the domain. In other words, $R$ or "is red" just refers to all of the red things.

Part of the definition of an interpretation, however, reads as follows: the interpretation "assigns sets of n-tuples of objects of $\mathcal{D}$ to n-place predicates." What is an n-place predicate? What is an n-tuple?

An n-place predicate is just a predicate term with $n$ number of places where we might fill-in names of objects. Or, it is a predicate term with $n$ number of places where it is necessary to fill-in a name so that the predicate becomes a proposition (something that can be true or false). So, for example, "$x$ is red" is a one-place predicate since there is one place (where the $x$ is) where if we were to fill-in a name of an object, the expression would express a proposition.

- "x is tall" is a 1-place predicate
- "x is taller than y" is a 2-place predicate
- "x is standing between y and z" is a 3-place predicate.

What is an n-tuple? Technically, an n-tuple refers to an ordered set with n elements. However, more simply, you might think of n-place predicates picking out different kinds of collections of objects. In the case of "x is red", when we interpret this predicate, we simply pick out all of the single red things. For example, this umbrella is red, that book is red, this pen is red, and so on. However, consider the two-place predicate "x is taller than y". In this case, the predicate doesn't just pick out tall objects. Instead, it picks out a collection of pairs of objects where the first object is taller than the second. Where each of the single objects picked out by "is red" is a 1-tuple, the pairs of objects are known as 2-tuples.

- "x is tall" is a 1-place predicate that is interpreted as a set of 1-tuples.
- "x is taller than y" is a 2-place predicate that is interpreted as a set of 2-tuples.
- "x is standing between y and z" is a 3-place predicate that is interpreted as a set of 3-tuples.

This covers the definition of a model. One final point concerns certain abbreviations for models, domains, and the interpretation of names and n-place predicates.

### 7.2.2 How to Recover a Model from a Completed Open Branch

In the previous section, I noted that a model can be constructed illustrating an argument's invalidity. Before doing this, we need a definition of validity and invalidity.

> Definition – Valid in **RL**
>
> An argument $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ therefore $\mathbf{Z}$ in **PL** is deductively valid if and only if there is no model $\mathcal{M}$ in which all of the members of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ are true and $\mathbf{Z}$ is false. A truth tree shows that an argument $P, Q, \ldots, Y \models Z$ if and only if $P, Q, R, \ldots, Y, \neg(Z)$ determines a closed tree.

> Definition – Invalid in **RL**
>
> An argument $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ therefore $\mathbf{Z}$ in **PL** is deductively valid if and only if there is at least one model $\mathcal{M}$ in which all of the members of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ are true and $\mathbf{Z}$ is false. A truth tree shows that an argument $P, Q, \ldots, Y \models Z$ if and only if $P, Q, R, \ldots, Y, \neg(Z)$ determines a completed open tree (a tree with at least one completed open branch).

In short, a **closed predicate logic tree** tells us that there is no model where the wffs in the stack are valuated as true. A **completed open tree** tells us that there is at least one model of the wffs in the stack where each wff is true.

Now suppose you set up a truth tree to test an argument to see if it was valid or invalid. Let's say that this argument is $P, Q$ therefore $Z$. You set the tree up by stacking $P, Q, \neg(Z)$, decompose the wffs in tree, and find that there is a completed open branch. The argument is therefore invalid. From the definition above, this implies that there is at least one model where $P$ and $Q$ are both true and $Z$ is false. How can I identify such a model?

Before stating this method, we need one more definition.

> Definition – Literal Wff in **RL**
>
> A literal wff (a literal) in **RL** is (i) an $n$-place predicate followed by $n$ names and (i) a negation followed by $n$-place predicate and $n$ names,

**Example 1: Some Examples of Literal Wffs**

1. $Pa$
2. $\neg Pa$
3. $Rab$
4. $Raa$
5. $\neg Raa$

From a completed open branch, we can recover a model in the following way:

1. Identify a completed open branch
2. From the base of the completed open branch, write down all of the literals in that branch.
3. Provide an interpretation for each name in every literal wff by assigning it one and only one item in the domain.
   - For $Pa$ and $\neg Qb$, $\mathscr{I}(a) = a$ and $\mathscr{I}(b) = b$
4. For every name interpreted in terms of an item in the domain, explicitly indicate that those items (objects) are a part of the domain.
   - Since $\mathscr{I}(a) = a$ and $\mathscr{I}(b) = b$, we specify that $\mathcal{D} = \{a, b\}$.
5. For each non-negated literal wff, provide an interpretation for each $n$-place predicate consistent with the truth of that literal wff.
   - For $Pa$, $Pb$, and $\neg Qb$, $\mathscr{I}(P) = \{a, b\}$ and $\mathscr{I}(Q) = \{a\}$ or $\mathscr{I}(P) = \{a, b\}$ and $\mathscr{I}(Q) = \{\}$

**Example 2: Consider $Pa, Rb, Lcc$ therefore $\neg(\forall x)Px$.**

| | | |
|---|---|---|
| 1 | $Pa$ | $P$ |
| 2 | $Rb$ | $P$ |
| 3 | $\neg Lc$ | $P$ |
| 4 | $\neg\neg(\forall x)Px$ | $P$ |
| 5 | $(\forall x)Px$ | $4\neg\neg D$ |
| 6 | $Pa$ | $5\forall D$ |
| 7 | $Pb$ | $5\forall D$ |
| 8 | $Pc$ | $5\forall D$ |

- $\mathcal{M} = \langle \mathcal{D}, \mathscr{I} \rangle$
- Since $Pc, Pb, Pa, \neg Lc$, and $Rb$ are literal wffs, we interpret each name in these wffs by assigning it an object in the domain $\mathcal{D}$ as follows: $\mathscr{I}(a) = a$, $\mathscr{I}(b) = b$, $\mathscr{I}(c) = c$.
- Since $\mathscr{I}(a) = a$, $\mathscr{I}(b) = b$, $\mathscr{I}(c) = c$, we indicate that each of these objects is in the domain: $\mathcal{D} = \{a, b, c\}$
- Since $Pa, Pb, Pc$, we interpret each of these objects as belonging to the interpretation $P$ as follows: $\mathscr{I}(P) = \{a, b, c\}$. Similarly, since $Rb$, $\mathscr{I}(R) = \{b\}$.

What the above model provides is a two-part configuration. It provides a representation of the world (the objects and n-place relations in the world) and an interpretation of **RL** that would make a set of wffs true. If the model provides an example of how all of the wffs could be true, then what it shows in the case above is an example where the premises $Pa$, $Pb$, $Rb$, and $\neg Lc$ are true and the conclusion $\neg(\forall x)Px$ is false. Thus, it gives a concrete illustration of how the argument is invalid.

Definition – Consistent

A set of propositions $\{P, Q, R, \ldots Z\}$ is consistent in **RL** if and only if there is at least one model where $P, Q, R, \ldots Z$ are true. A truth tree shows that $\{P, Q, R, \ldots, Z\}$ is consistent if and only if a complete tree of the stack of $P, Q, R, \ldots, Z$ determines a completed open tree. That is, if there is at least one completed open branch.

Definition – Inconsistent

A set of propositions $\{P, Q, R, \ldots Z\}$ is inconsistent in **RL** if and only if there is no model where $P, Q, R, \ldots Z$ are true. A truth tree shows that $\{P, Q, R, \ldots, Z\}$ is inconsistent if and only if a tree of the stack of $P, Q, R, \ldots, Z$ is a closed tree. That is, if all branches close.

Definition – semantic consequence (entailment)

A closed wff **Q** is a *semantic consequence* (entailment) in **PL** of a set of closed wffs $\Gamma$ if and only if there is no model $\mathcal{M}$ in which all of the members of $\Gamma$ are true and **Q** is false. A truth tree shows that $\Gamma \models \mathbf{Q}$ if and only if $\Gamma \wedge \neg\mathbf{Q}$ determines a closed tree.

DEFINITION – VALID IN **RL**

An argument $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ therefore **Z** in **PL** is deductively valid if and only if **Z** is a semantic consequence of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$. That is, $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y}$ therefore **Z** is deductively valid if and only if $\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Y} \models \mathbf{Z}$. A truth tree shows that an argument $P, Q, \ldots, Y \models Z$ if and only if $P, Q, R, \ldots, Y, \neg(Z)$ determines a closed tree.

**Exercise 2:** *Determine whether the following sets of* **RL** *wffs are consistent or inconsistent. If consistent, construct a model that illustrates that the set of wffs is consistent.*

1. $(\forall x)\neg Px, (\exists x)Px$
2. $Pa, Pb, (\exists x)Px \wedge (\forall x)Px, (\exists x)Px$
3. $(\forall y)(Py \vee Ry), (\exists x)(\neg Px \wedge \neg Rx)$

**Exercise 3:** *Determine whether the following arguments are valid or invalid. If the argument is invalid, construct a model that illustrates that the argument is invalid.*

1. $(\exists x)Mx$ therefore $Pa$
2. $(\exists x)Mx \wedge (\exists x)Rx$ therefore $(\exists x)(Mx \wedge Rx)$
3. $(\forall x)(Zx \rightarrow Mx)$ therefore $(\exists x)Zx$

**QUESTIONS**

1. What does $\Gamma \models Q$ mean?
2. Know the decomposition rules for **PL** and **RL**.
3. Know how to setup and decompose a tree for a set of wffs and for an argument.
4. Under what conditions does a truth tree show that an argument is invalid?
5. Under what conditions does a truth tree show that an argument is valid?
6. Under what conditions does a truth tree show that a set of wffs is consistent?
7. Under what conditions does a truth tree show that a set of wffs is inconsistent?
8. Be able to recover a model from a completed open branch.