lecture 10, trees for PL

phil1012 introductory logic

overview

this lecture

- an introduction to **truth trees** for PL
- general motivation for the use of truth trees for PL
- particular motivations for each tree rule

next lecture

- how to construct truth trees in general
- how to use truth trees to test for particular logical properties

learning objectives

- after doing the relevant reading for this lecture, listening to the lecture, and attending the relevant tutorial, you will be able to:
  - explain the motivation behind truth trees for PL
  - write down the tree rules for PL from memory

required reading

- all of chapter 7

the motivation for truth trees

the motivation for truth trees

- the limitations of truth tables as a proof method
- proof trees are the only method of proof we will examine for MPL, GPL, and GPLI

the limitations of truth tables as a proof method

- recall the maths concerning truth tables
  - 2 basic propositions: 4 rows
  - 3 basic propositions: 8 rows
  - 4 basic propositions: 16 rows
  - 5 basic propositions: 32 rows
  - 6 basic propositions: 64 rows
  - . . .

the limitations of truth tables as a proof method

- truth tables (or complete ones at least) involve a kind of enumerative search
- if an argument is valid, you've got to fill in every row to be sure
- truth tables may eventually provide an evaluation which is a counterexample
- might we work in a more direct fashion towards a counterexample?
- yes: truth trees

the basic idea behind truth trees

re-inventing truth trees

- let's re-invent truth trees
- suppose you have some proposition or other, and you want to know whether it is satisfiable

$((A \rightarrow B) \wedge (A \wedge \neg B))$

- suppose we begin by assuming that it *is* satisfiable
- that is, suppose there is some assignment on which it is true
- what do we know?
- well, we know that the assignment is one on which both conjuncts must be true
- for now, let's just write out a list like this to keep track:

$((A \rightarrow B) \wedge (A \wedge \neg B))$, $(A \rightarrow B)$, $(A \wedge \neg B)$

- okay, now we've got another conjunction in there.
- if there is an assignment on which all these propositions are true together, it must be an assignment on which both conjuncts are true
- so let's expand our list:

$((A \rightarrow B) \wedge (A \wedge \neg B))$, $(A \rightarrow B)$, $(A \wedge \neg B)$, A, $\neg B$

- okay, now comes the tricky part. how do we deal with the conditional?
- well, let's ask what kind of assignment we need in order to make it true?
- here it helps to remind ourselves that $(A \rightarrow B)$ is equivalent to $(\neg A \vee B)$
- so we can either take an assignment on which $\neg A$ is true, or an assignment on which B is true
- take the first first

$((A \rightarrow B) \wedge (A \wedge \neg B))$, $(A \rightarrow B)$, $(A \wedge \neg B)$, A, $\neg B$, $\neg A$

- uh oh. what is wrong with this list of propositions?

$((A \rightarrow B) \wedge (A \wedge \neg B))$, $(A \rightarrow B)$, $(A \wedge \neg B)$, A, $\neg B$, $\neg A$

- not all of the propositions can be true at once! A and $\neg A$ cannot be true at once.
- this path isn't going to make our initial assumption about satisfiability work out

---

- what about the other assignment, on which B is true?

$((A \to B) \land (A \land \neg B)) , (A \to B) , (A \land \neg B)), A , \neg B , B$

- uh oh. not all of the propositions can be true at once! B and $\neg B$ cannot be true at once

---

- our initial assumption was mistaken. the proposition is not satsfiable!
- this method looks like a good one for refuting the initial assumption of satisfiability
- if we reach a contradiction on a path, we can reject that path
- if we reach a contradiction on all paths, we know the proposition isn't satisfiable

---

- what if we don't read a contradiction?
- what if had started with a slightly different proposition?
- what if we had arrived at the following after splitting the conditional?

$((C \to B) \land (A \land \neg B)) , (C \to B) , (A \land \neg B)), A , \neg B , \neg C$

- well, assuming that we have broken down every proposition once
- we know that all of the propositions can be true together if and only if the basic propositions and negations of basic propositions can be true together
- and we can just see that A, $\neg B$ and $\neg C$ can all be true together
- so we can conclude that the original proposition is satisfiable

---

- even better
- we have derived an assignment on which the proposition is true:

  A: T, B: F, C: F

- unlike truth tables, truth trees can take us directly to the relevant assignment

---

- essential to the procedure is the idea of "splitting" or "branching"
- we must check *every* path we create after splitting or branching
- else we won't know if there is *some* path without a contradiction on it

---

- drawing this method as a tree going down the page is even better

the tree method

the tree method described

- here is the tree proof of the unsatisfiability of $((A \to B) \land (A \land \neg B))$ from above

$((A \to B) \wedge (A \wedge \neg B))\checkmark$
$(A \to B)\checkmark$
$(A \wedge \neg B)\checkmark$
A
$\neg B$
$\neg A \quad B$
x x

- how do we construct such a tree?
- well, we begin by writing down the proposition (or propositions) we want to test for satisfiability

$((A \to B) \wedge (A \wedge \neg B))$

- think of this proposition as the "root" of the tree if you like (the tree is upside down)

- now we write down other things which must be true assuming these are true (according to the tree rules)
- if it follows from the assumption that α is true that β and γ are true, we write β and γ at the bottom of every open path on our tree
- in this case we only have one path

$((A \to B) \wedge (A \wedge \neg B))$
$(A \to B)$
$(A \wedge \neg B)$

- in a moment, we will discuss the rules for building trees (the tree rules)
- these just encapsulate our claims about "what must be true" if some proposition is true
- each time we apply a tree rule to a proposition we "check it off" with a tick
- this is just a way of keeping track what we've "dealt with" that proposition

$((A \to B) \wedge (A \wedge \neg B))\checkmark$
$(A \to B)$
$(A \wedge \neg B)$

- here is the step applied to the conjunction

$((A \to B) \wedge (A \wedge \neg B))\checkmark$
$(A \to B)$
$(A \wedge \neg B)\checkmark$
A
$\neg B$

- if it follows from the assumption that α is true that either β and γ are true, or both are true, then we "split" the tree and write them down on either side of the new branch
- we split on every open path the proposition appears on

$((A \to B) \wedge (A \wedge \neg B))\checkmark$
$(A \to B)\checkmark$
$(A \wedge \neg B)\checkmark$
A
$\neg B$
$\neg A \quad B$

- note that each time we apply a rule, we write down propositions which are simpler than those we began with
- the process ends when we cannot apply any more rules
- whenever we encounter a contradiction on a path we immediately "close" the path/branch with a cross
- there's no point continuing with that path

- in the case at hand, both paths close

$((A \to B) \wedge (A \wedge \neg B))\checkmark$
$(A \to B)\checkmark$
$(A \wedge \neg B)\checkmark$
A
$\neg B$
$\neg A \quad B$
x x

- because propositions always get simpler, we eventually end up with only basic propositions and negations of basic propositions
- if a path contains a contradiction, the propositions on that path cannot all be true at once
- if a finished path does not contain a contradiction, the propositions can all be true together
- we can "read off" an assignment on which all the propositions are true from the basic propositions and negated propositions on such a path

the tree rules

the tree rules stated

- rather than thinking about "what must be true" every time we build a tree, we can write down some rules that capture the relevant claims for each connective
- we'll go over the motivation here
- if you forget a tree rule, it is helpful to know the motivation
- if is also helpful to think of the truth table for the connectives (and connectives in the immediate scope of negation).

disjunction

- here is the tree rule for disjunction:

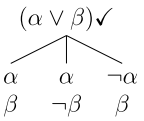
$(\alpha \vee \beta)\checkmark$
$\alpha \quad \beta$

- let's think through the motivation for the rule in terms of truth tables

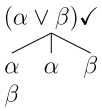
- ask: what must be *true* if the disjunction is true?

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

- reading off from the rows on which $(\ \alpha \vee \beta)$ is true . . .



- then simplifying . . .

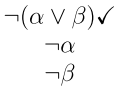


- and simplifying again . . .

- what must be *true* if the disjunction is true?
- answer: either α *or* β
- the tree rule for disjunction is easy to remember even though there's a bit involved in reading the tree rule off the truth table

negated disjunction

- here is the tree rule for negated disjunction:



- let's think through the motivation for the rule in terms of truth tables
- ask: what must be *true* if the negated disjunction is true?

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

- answer: $\neg\alpha$ *and* $\neg\beta$ must be true
- the tree rule for negated disjunction is easy to read off its truth table

conjunction

- here is the tree rule for conjunction:

$(\alpha \wedge \beta)$✓
α
β

- let's think through the motivation for the rule in terms of truth tables

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

- the tree rule for conjunction is easy to read off its truth table

negated conjunction

- here is the tree rule for negated conjunction:

$\neg(\alpha \wedge \beta)$✓
$\neg\alpha \quad \neg\beta$

- let's think through the motivation for the rule in terms of truth tables

α	β	$\neg(\ \alpha \wedge \beta)$
T	T	F
T	F	T
F	T	T
F	F	T

- reading off the tree rule for negated conjunction works a bit like the case for disjunction . . .

$\neg(\alpha \wedge \beta)$✓
$\alpha \quad \neg\alpha \quad \neg\alpha$
$\neg\beta \quad \beta \quad \neg\beta$

- then simplifying . . .

$\neg(\alpha \wedge \beta)$✓
$\neg\beta \quad \neg\alpha \quad \neg\alpha$
$\neg\beta$

- then simplifying again . . .

$\neg(\alpha \wedge \beta)$✓
$\neg\alpha \quad \neg\beta$

conditional

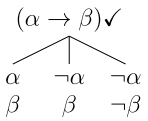
- here is the tree rule for conditional:

$(\alpha \rightarrow \beta)$✓
$\neg\alpha \quad \beta$

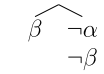
- let's think through the motivation for the rule in terms of truth tables

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

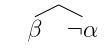
- reading the tree rule of the truth table for conditional is a little trikier than the others . . .
- we start with . . .



- simplifying first on β we get . . .



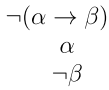
- then simplifying on $\neg\beta$ . . .



- in the case of conditional it is probably easier to just remember that $(\ \alpha \rightarrow \beta)$ is equivalent to $(\ \neg\alpha \vee \beta)$
- the tree rule (and the motivaiton for the rule) will then be obvious!

negated conditional

- here is the tree rule for negated conditional:

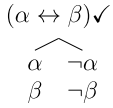


- let's think through the motivation for the rule in terms of truth tables

α	β	$\neg(\ \alpha \rightarrow \beta)$
T	T	F
T	F	T
F	T	F
F	F	F

biconditional

- here is the tree rule for biconditional:



- let's think through the motivation for the rule in terms of truth tables

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

negated biconditional

- here is the tree rule for negated biconditional:

$\neg(\alpha \leftrightarrow \beta)$✓

α $\quad$ $\neg\alpha$

$\neg\beta$ $\quad$ β

- let's think through the motivation for the rule in terms of truth tables

α	β	$\neg(\ \alpha \leftrightarrow \beta)$
T	T	F
T	F	T
F	T	T
F	F	F

double negation

- here is the tree rule for double negation:

$\neg\neg\alpha$✓

α

- let's think through the motivation for the rule

α	$\neg\neg\alpha$
T	T
F	F

closure

- finally, here is the closure rule:

α

|

...

|

$\neg\alpha$

x

- let's think through the motivation for the rule
- if a proposition and its negation both appear on a path, then it is not the case that all the propositions on that path can be true at the same time
- so we close that path

- a note on closure: the closure rule applies to *all* propositions and

their negations, not just basic propositions and their negations

wrapping up

this lecture

- an introduction to **truth trees** for PL
- general motivation for the use of truth trees for PL
- particular motivations for each tree rule

next lecture

- how to construct truth trees in general
- how to use truth trees to test for particular logical properties

next lectures

- lecture 11, uses of trees for PL
- lecture 12, issues in translation: conditional
- lecture 13, the formal language MPL