# ELEMENTS OF DEDUCTIVE LOGIC

15. Predicate Logic: tableau methods

J. Chandler

*KUL 2012*





## Introduction

- Last time: wrapping up the semantics for $\mathcal{L}_P$ + introducing tableaux
  - Truth in a model: the general case
  - Truth in *finite* models
  - Testing for validity: a special case
  - Testing for validity: the general case (tbc)
- This time: more on the last point, i.e. tableaux





## Universal formulae

- The rule:

$$\begin{array}{c}(\forall x)A\\ \vert\\ A(x := a)\end{array}$$

($a$ already on branch)

- Rationale:
  - If $(\forall x)A$ is true, then all of its instances are true.
  - This includes all those instances involving names already used on the branch.





## Negated universal formulae

- The rule:

$$\begin{array}{c}\sim(\forall x)A\\ \vert\\ \sim A(x := a)\end{array}$$

($a$ new)

- Rationale:
  - If $\sim(\forall x)A$ is true, then $(\forall x)A$ is false, so not all of the instances of the latter are true, so at least one of them is false and hence its negation is true.
  - This might not be an instance that involves a name already used on the branch. So we introduce a new name, just in case.

## Particular vs general rules

- The rules for $(\exists x)A$ and $\sim(\forall x)A$ are known as particular rules:
  They are applied only *once* to a given formula.
- Once used, we put a tick next to the formula alongside the name introduced: e.g. $\checkmark a$
- The rules for $\sim(\exists x)A$ and $(\forall x)A$ are known as general rules:
  They can be applied *repeatedly* to one same formula
- After first use, we put a backslash next to the formula alongside the relevant name: e.g. $\backslash a$.
- Upon subsequent uses, we simply add the relevant namese.g. $\backslash a, b, c \ldots$





## Note on general rules

- Question:
  What happens when all the formulae at the root of the tableau are of the form $\sim(\exists x)A$ or $(\forall x)A$ and do not contain any names?
- Answer:
  - Since every domain has at least one element and every element is named, we have to have at least one name by default, say '$a$'.
  - We then substitute this into all the relevant formulae, using the relevant rules.





## Note on general rules (ctd.)

- We show that $(\forall x)Gx \vdash (\exists x)Gx$.

$$
\begin{array}{c}
(\forall x)Gx\ \backslash a \\
\sim(\exists x)Gx\ \backslash a \\
| \\
Ga \\
| \\
\sim Ga \\
\times
\end{array}
$$





## Completedness

- Note: there are no repeatable rules in propositional logic.
- The introduction of these rules makes a difference to the definition of a completed tree in pred. logic.
- In prop. logic:
  A tree is completed iff, in every open branch $b$, every formula on $b$ that could have had a rule applied to it has had a rule applied to it.
- This is not good enough here: some formulae sometimes need to have rules applied to them more than once.
- In pred. logic:
  A tree is completed iff, in every open branch $b$, (i) every formula on $b$ that could have had a rule applied to it has had a rule applied to it *and* (ii) every name on $b$ has been substituted into every formula of the form $\sim(\exists x)A$ or $(\forall x)A$.

## Brief general tips

- Apply propositional rules first, starting with non-branching rules.
- Then apply instantion rules, starting with particular rules.
- These recommendations are *defeasible*, however: e.g. the application of a general rule may immediately lead to tableau closure.
- Now for 3 examples: a tableau that closes, an open tableau with countermodel and a little surprise...





## A closed tableau

- We show that $(\forall x)(Fx \supset Gx), (\exists x) \sim Gx \vdash (\exists x) \sim Fx$

$$
\begin{array}{c}
(\forall x)(Fx \supset Gx)\ \backslash a \\
(\exists x) \sim Gx\ \checkmark a \\
\sim (\exists x) \sim Fx\ \backslash a \\
| \\
\sim Ga \\
| \\
Fa \supset Ga\ \checkmark \\
| \\
\sim\sim Fa \\
\diagup\ \ \diagdown \\
\sim Fa \quad Ga \\
\times \quad\ \times
\end{array}
$$





## A completed open tableau

- We show that $(\exists x)Fx, (\exists x)Gx \nvdash (\exists x)(Fx \& Gx)$

$$
\begin{array}{c}
(\exists x)Fx\ \checkmark a \\
(\exists x)Gx\ \checkmark b \\
\sim (\exists x)(Fx \& Gx)\ \backslash a,b \\
| \\
Fa \\
| \\
Gb \\
| \\
\sim (Fa \& Ga)\ \checkmark \\
\diagup\ \ \diagdown \\
\sim Fa \quad \sim Ga \\
\times \quad\ \ | \\
\quad\ \ \ldots
\end{array}
\qquad
\begin{array}{c}
\ldots \\
| \\
\sim (Fb \& Gb)\ \checkmark \\
\diagup\ \ \diagdown \\
\sim Fb \quad \sim Gb \\
\uparrow \quad\ \times
\end{array}
$$





## A completed open tableau: the countermodel

- Our open branch contains: $Fa$, $Gb$, $\sim Ga$ and $\sim Fb$.
- So we have a domain $D$ of size 2, say $D = \{d, e\}$, with $I$ defined as follows:

| $I$ | |
|---|---|
| $a$ | $d$ |
| $b$ | $e$ |

| $I(F)$ | |
|---|---|
| $d$ | 1 |
| $e$ | 0 |

| $I(G)$ | |
|---|---|
| $d$ | 0 |
| $e$ | 1 |

## An interminable open tableau (!)

- Yes indeed.
- We check whether or not $\vdash\sim (\forall x)(\exists y)Lxy$

$$
\begin{array}{l}
(\forall x)(\exists y)Lxy \;\backslash a_1, a_2, a_3 \\
\quad | \\
(\exists y)La_1y \;\checkmark a_2 \\
\quad | \\
La_1a_2 \\
\quad | \\
(\exists y)La_2y \;\checkmark a_3 \\
\quad | \\
La_2a_3 \\
\quad | \\
(\exists y)La_3y\ldots
\end{array}
$$





## An interminable open tableau (!) (ctd.)

- What's going on here?
- It turns out that there *exists* a proof of invalidity: the completed tableau is in fact open.
- Problem: the completed open tableau is infinitely long!
- So we *cannot find* this proof in a finite number of steps by sequentially applying rules until we complete the tableau.
- This is also demonstrably true of any other mechanisable procedure for finding a proof of either validity or invalidity in pred. logic.
- We say that predicate logic is undecidable.
- This is *not* the case for propositional logic, nor is it the case for the restriction of predicate logic to monadic predicates.
- In both those cases, completed trees are always finite.



## Next session

- Tableau exercises: check Toledo.
- Session after that: finishing off tableaux + identity and definite descriptions.
- Reading: Restall Ch. 11.