# DECISION PROBLEMS IN PROPOSITIONAL/PREDICATE LOGIC

M.Lupea

1. Is a propositional/predicate formula a ***tautology/theorem***?

$$\overset{?}{\models} V \quad \text{or} \quad \overset{?}{\vdash} V$$

2. Is a propositional/ predicate formula a ***logical/syntactic consequence*** of a set of hypotheses?

$$U_1,\ldots,U_n \overset{?}{\models} V \quad \text{or} \quad U_1,\ldots,U_n \overset{?}{\vdash} V$$

In order to solve these two decision problems, ***theorem proving methods*** are applied.

# Classification:

## semantic *versus* syntactic proof methods

| *semantic proof methods* | *syntactic proof methods* |
|---|---|
| **propositional logic**<br>• the truth table method<br>• CNF- conjunctive normal form<br><br>**propositional/predicate logic**<br>• the semantic tableaux method | **propositional/predicate logic**<br>• the definition of deduction<br>• the theorem of deduction and its reverse<br>• the resolution method<br>• the sequent calculus method |

## Classification:

### direct *versus* refutation proof methods

| **direct** ***proof methods*** | **refutation** ***proof methods*** |
| --- | --- |
| -they use directly the formula to be proved | - they model "**reductio ad absurdum**" principle (proof by contradiction) using the negation of the formula to be proved |
| **propositional logic**<br>• the truth table method<br>• CNF- conjunctive normal form<br>**propositional/predicate logic**<br>• the sequent calculus method<br>• the definition of deduction<br>• the theorem of deduction and its reverse | **propositional/predicate logic**<br>• the semantic tableaux method;<br>• the resolution method. |

M.Lupea

# Semantic Tableaux Method

- It was proposed as a proof method for classical logics by R. Smullyan in 1968.
- Dedicated theorem provers: 3TAP, pTAP, leanTAP, Cassandra.
- It was easily adapted to *nonstandard logics* (modal, temporal, many-valued, non-monotonic).
- It is based on semantic considerations => ***semantic method.***
- Its basic aim is to decide ***consistency*** and to find all the models of a formula by decomposing the formula in subformulas.
- The ***validity*** of a formula **is proved by contradiction**=>

  => ***refutation method.***

M.Lupea

# Decomposition rules for *propositional formulas*

➢ **conjunctive formulas**, which are consistent only if both of its component sub-formulas are satisfied, are decomposed using $\alpha$ - **rules**

$$
\begin{array}{ccc}
A \wedge B & \neg(A \vee B) \equiv \neg A \wedge \neg B & \neg(A \to B) \equiv A \wedge \neg B \\
| & | & | \\
A & \neg A & A \\
| & | & | \\
B & \neg B & \neg B
\end{array}
$$

➢ **disjunctive formulas**, which are satisfied if one of its component sub-formulas is satisfiable(consistent) are decomposed using $\beta$ - **rules**

$$
\begin{array}{ccc}
A \vee B & \neg(A \wedge B) \equiv \neg A \vee \neg B & A \to B \equiv \neg A \vee B \\
/ \quad \backslash & / \quad \backslash & / \quad \backslash \\
A \quad\quad B & \neg A \quad\quad \neg B & \neg A \quad\quad B
\end{array}
$$

**Recommendation:**

- apply $\alpha$ - rules which keep one branch before $\beta$ - rules which make a branching.

# Decomposition rules for *predicate formulas*

***γ - rules***

$$
\begin{array}{ccc}
(\forall x)A(x) & \quad & \neg(\exists x)A(x) \\
| & & | \\
A(c_1) & & \neg A(c_1) \\
| & & | \\
A(c_2) & & \neg A(c_2) \\
| & & | \\
\ldots & & \ldots \\
| & & | \\
A(c_n) & & \neg A(c_n) \\
| & & | \\
(\forall x)A(x) & & \neg(\exists x)A(x)
\end{array}
$$

- used to decompose **universally quantified** formulas
- $c_1, c_2, \ldots, c_n$ are all the parameters (constants) on that branch and they are used for the instantiation of $A$.
- if there is no constant on that branch, a new constant is introduced and used for instantiation.

***δ - rules***

$$
\begin{array}{ccc}
(\exists x)A(x) & \quad & \neg(\forall x)A(x) \\
| & & | \\
A(c) & & \neg A(c)
\end{array}
$$

- used to decompose **existentially quantified** formulas
- ***c*** is a new constant on the branch.

**<u>Recommendation:</u>**

- apply $\delta$-rules (introduction of new constants) before $\gamma$-rules which use all the constants on that branch.

# Construction of a semantic tableau

M.Lupea

To a propositional/predicate formula $U$ we can associate a ***semantic tableau***, which is a binary tree having formulas in its nodes and it is built as follows:

1. the root of the tree is labeled with the initial formula;
2. every branch of the tree which contains a formula will be extended with a subtree according to the decomposition rule specific to its class;
3. the extension of a branch stops in the following cases:
   a) if that branch contains a formula and its negation, the branch is marked as closed using the symbol $\otimes$;
   b) if all the formulas on that branch are already decomposed or if by decomposing the formulas which are not decomposed yet, no new formulas are obtained.

# Definitions

1. A ***branch*** of a semantic tableau is called ***closed*** (marked by $\otimes$ ) if it contains a formula and its negation, otherwise the ***branch*** is called ***open*** (marked by the symbol $\odot$).
2. A ***branch*** is called ***complete*** if it is closed or all the formulas on that branch are already decomposed.
3. A ***semantic tableau*** is called ***closed*** if all its branches are closed.
4. If a semantic tableau has at least one open branch, the ***tableau*** is called ***open***.
5. A ***semantic tableau*** is called ***complete*** if all its branches are complete.

**Remarks:**

- A ***closed branch*** symbolizes ***inconsistency*** among the formulas on that branch.
- An ***inconsistent formula*** has associated a ***closed semantic tableau***.
- The ***set of formulas belonging to a complete and open branch is consistent***, meaning that it has a model.
- A ***consistent formula*** has associated a ***complete and open semantic tableau***.

# Models of a formula

M.Lupea

- ***propositional formula*** $U$ **:**
  - by assigning the truth value $T$ to all the literals belonging to a complete and open branch of the semantic tableau of $U$, a partial/complete model of $U$ is obtained.
  - if a propositional variable is not on that branch, we can assign to it either $T$ or $F$, obtaining complete models.

- ***predicate formula*** $U$ **:**
  - an open branch of the semantic tableau associated to *formula* $U$ provides a partial/complete model of $U$ with the minimum number of elements in the domain of interpretation. Such a model $I = < D, m >$ is a 'generic' one and it is built as follows:
    - the domain of interpretation $D$ contains all the constants on that branch;
    - the value $T$ is assigned to all the literals on that branch:

      if $c_1, \ldots, c_n \in D$, $P \in \mathrm{P}_n$ ($P$ is an $n$ - ary predicate symbol) and

      $$\begin{cases} \text{a) } P(c_1, \ldots, c_n) \text{ belongs to the branch then } m(P)(c_1, \ldots, c_n) = T \\ \text{b) } \neg P(c_1, \ldots, c_n) \text{ belongs to the branch then } m(P)(c_1, \ldots, c_n) = F \end{cases}$$
  - based on this generic model, more concrete models can be obtained.

# Theorems

Semantic tableaux method – a *refutation* proof method

**Theorem 1: Soundness and completeness of the semantic tableaux method**

**A propositional/predicate formula $U$ is a tautology *if and only* if $\neg U$ has a closed semantic tableau.**

**Theorem 2:**

**Let $U_1, U_2, \ldots, U_n, V$ be propositional/predicate formulas.**

$U_1, U_2, \ldots, U_n \models V$ *if and only if* **there is a closed semantic tableau associated to the formula** $U_1 \wedge U_2 \wedge \ldots \wedge U_n \wedge \neg V$.

**Propositional logic is decidable:**

- the ***semantic tableau*** associated to a propositional formula ***is always finite*** therefore, we can always decide whether the formula is a tautology or not.

# Predicate logic - undecidable

**Theorem** (Church 1936):

The problem of validity of a first-order formula is ***undecidable***, but is ***semi-decidable***.

If a procedure ***Proc*** is used to check the validity of a first-order formula ***U*** we have the following cases:

- **if** $U$ is a valid formula, **then** ***Proc*** ends with the corresponding answer.
- **if** the formula $U$ is not valid, **then** ***Proc*** ends with the corresponding answer **or** ***Proc*** may never stop.

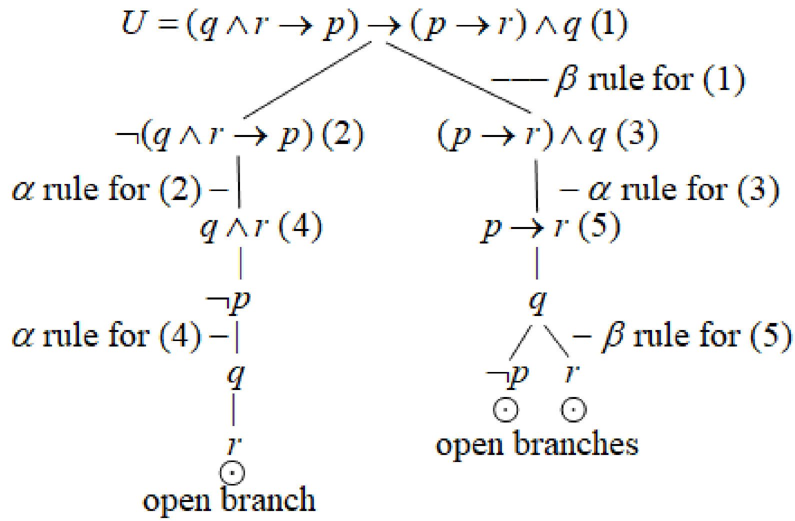
**Predicate logic is not decidable, it is only semi-decidable**:

- if the ***semantic tableau*** associated to the predicate formula $\neg U$ is ***finite*** then we can decide whether the formula $U$ is a ***tautology*** (closed tableau for $\neg U$) or not (complete and open tableau for $\neg U$);
- if the ***semantic tableau*** of the predicate formula $\neg U$ is ***infinite*** then we cannot decide upon the validity of $U$.

M.Lupea

## **Example 1**

Build a semantic tableau for $U = (q \wedge r \to p) \to (p \to r) \wedge q$ and decide the type of $U$. In case of consistency find all its models.




The semantic tableau is complete and open, having three open branches and thus U is consistent. $\text{DNF}(U) = (\neg p \wedge q \wedge r) \vee (q \wedge \neg p) \vee (q \wedge r) \equiv (q \wedge \neg p) \vee (q \wedge r)$

The simplified form of DNF was obtained by applying the absorption law. The left-most branch, corresponding to the first cube, is covered by the other two branches.

| | |
|---|---|
| The branch represented by the cube $q \wedge \neg p$ provides the models:<br>$i_1, i_2 : \{p, q, r\} \to \{T, F\}$<br>$i_1(p) = F, i_1(q) = T, i_1(r) = T$<br>$i_2(p) = F, i_2(q) = T, i_2(r) = F$ | The branch represented by the cube $q \wedge r$ provides the models:<br>$i_3, i_4 : \{p, q, r\} \to \{T, F\}$<br>$i_3(p) = T, i_3(q) = T, i_3(r) = T$<br>$i_4(p) = F, i_4(q) = T, i_4(r) = T$ |

Because $i_1 = i_4$, $U$ has three models: $i_1$, $i_2$ and $i_3$, $i_1(U) = i_2(U) = i_3(U) = T$

Having at least one model, $U$ is a consistent formula, but it is not a tautology (does not have 8 models), therefore *$U$ is a contingent formula.*

# Example 2



Using the semantic tableaux method check whether the conclusion $C$ is a logical consequence of the set of hypotheses: $\{H_1, H_2, H_3, H_4\}$.

Consider the following ***hypotheses***:

$H_1$. Mary will go to London this summer if both her friends Kate and Susan go.

$H_2$. If Kate passes the English exam in May then she will go to London.

$H_3$. Kate was in hospital from April until July and she didn't take the English exam.

$H_4$. This summer Susan will go to London on a business trip.

and the ***conclusion***:

$C$. Will Mary go to London this summer?

Notations for the propositional variables:

$M$ – Mary will go to London
$S$ – Susan will go to London
$K$ – Kate will go to London
$KE$ – Kate passed the English exam

$H_1 : K \wedge S \rightarrow M$

$H_2 : KE \rightarrow K$

$H_3 : \neg KE$

$H_4 : S$

$C : M$

# Example 2

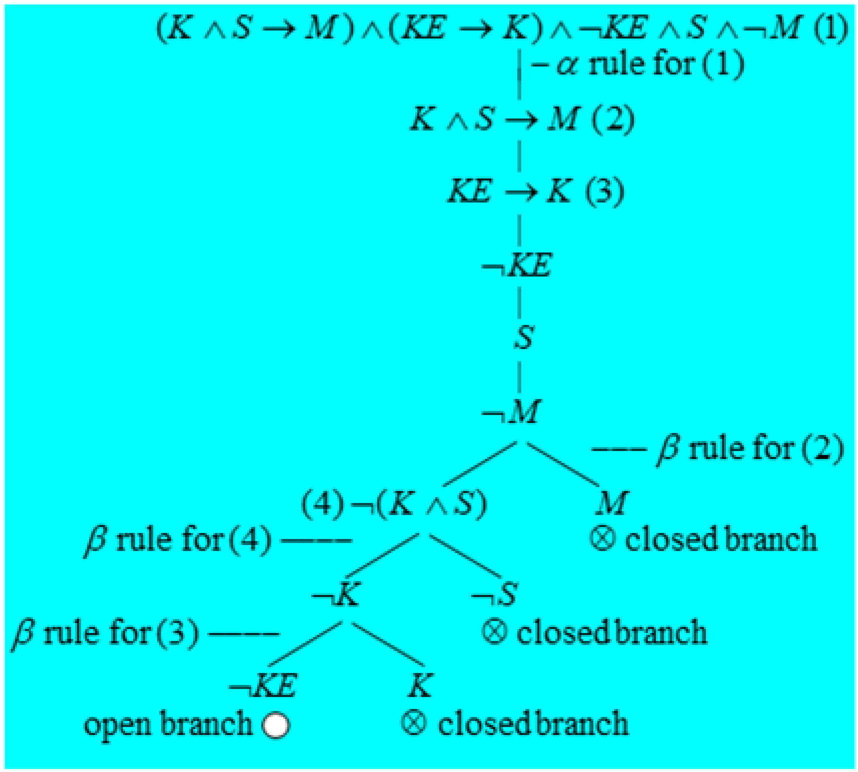
M.Lupea

The semantic tableau method is a refutation proof method, thus we have to negate the conclusion and use the Theorem of soundness and completeness:

$H_1, H_2, H_3, H_4 \models C$ *if and only if*

$H_1 \wedge H_2 \wedge H_3 \wedge H_4 \wedge \neg C$ has a closed semantic tableau.

The semantic tableau corresponding to the conjunction of the hypotheses and the negation of the conclusion is depicted below.




We have obtained a complete and open tableau with one open branch (the leftmost one) and three closed branches containing the following pairs of opposite literals: $(\neg K, K)$, $(S, \neg S)$, $(M, \neg M)$.

Therefore $H_1, H_2, H_3, H_4 \not\models C$ and based on the hypotheses we can't conclude that *'Mary will go to London this summer'*.

M.Lupea

## Example 3.

Prove the validity of $U = (\neg q \to \neg p) \to ((\neg q \to p) \to q)$

We build a semantic tableau of $\neg U$.

$$\neg U = \neg((\neg q \to \neg p) \to ((\neg q \to p) \to q)) \; (1)$$

| – $\alpha$ rule for (1)

$\neg q \to \neg p$ (2)

|

$\neg((\neg q \to p) \to q)$(3)

| – $\alpha$ rule for (3)

$\neg q \to p$ (4)

|

$\neg q$

/ \ —— $\beta$ rule for (2)

$q$ $\qquad$ $\neg p$

closed branch $\otimes$

/ \ —— $\beta$ rule for (4)

$q$ $\qquad$ $p$

closed branch $\otimes$ $\qquad$ $\otimes$ closed branch

All three branches of the semantic tableau are closed, containing pairs of opposite literals: $(\neg q, q)$, $(\neg q, q)$, $(\neg p, p)$.

Therefore the formula $\neg U$ has no models so, it is an inconsistent formula.

We conclude that $U$ is a tautology.

**$\alpha$ - rule**

$\neg(A \to B)$

|

$A$

|

$\neg B$

**$\beta$ - rule**

$A \to B$

/ \

$\neg A$ $\qquad$ $B$

**Example 4.** Build two different semantic tableaux for the formula:

$$U = (p \vee q) \wedge \neg(q \rightarrow r) \wedge (p \rightarrow q \wedge r).$$

M.Lupea

**Tableau 1**

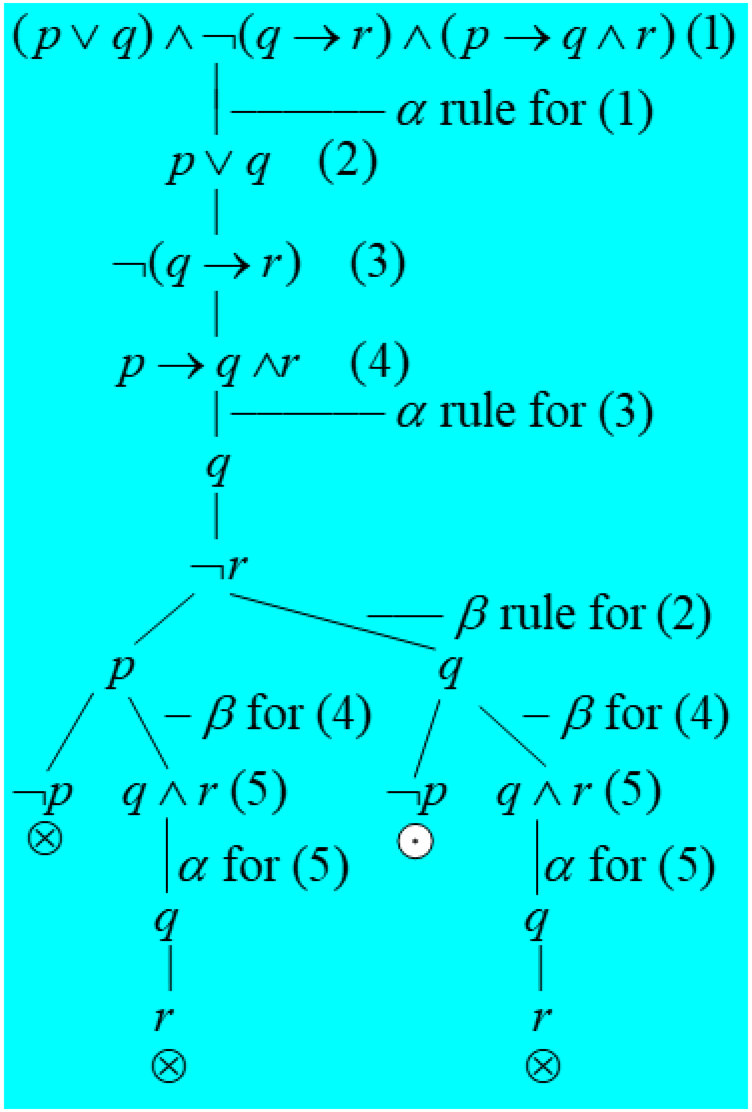
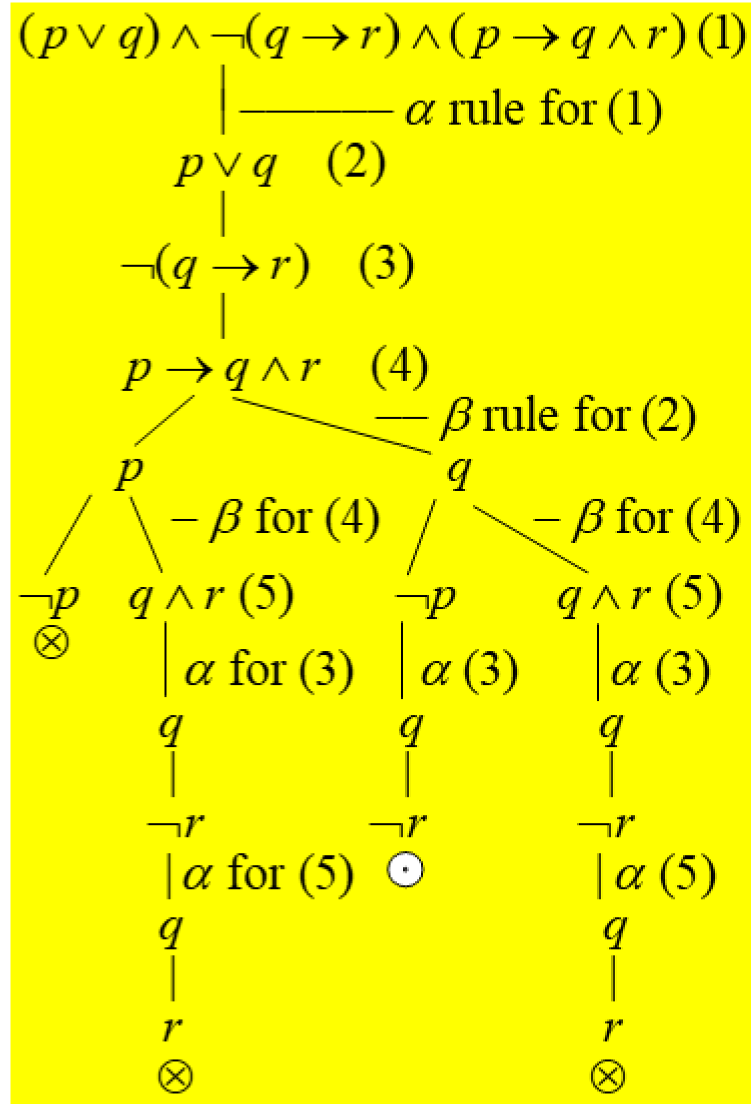



**Tableau 2**



# Example 5:

Check the **distributivity property of the** ***universal quantifier*** over ***implication***:

$$(\forall x)(P(x) \to Q(x)) \equiv (\forall x)P(x) \to (\forall x)Q(x)$$

We have that:

$(\forall x)(P(x) \to Q(x)) \equiv (\forall x)P(x) \to (\forall x)Q(x)$ if and only if

$\models (\forall x)(P(x) \to Q(x)) \leftrightarrow ((\forall x)P(x) \to (\forall x)Q(x))$ if and only if

$\models U_1$ and $\models U_2$ where:

$U_1 = (\forall x)(P(x) \to Q(x)) \to ((\forall x)P(x) \to (\forall x)Q(x))$ and

$U_2 = ((\forall x)P(x) \to (\forall x)Q(x)) \to (\forall x)(P(x) \to Q(x))$.

We shall prove that $U_1$ is a tautology and $U_2$ is not valid building the semantic tableaux of $\neg U_1$ and $\neg U_2$.

The following **theoretic result** is used:

$\models U$ ***if and only if*** $\neg U$ **has a closed semantic tableau.**

M.Lupea

$\neg U_1 = \neg((\forall x)(P(x) \to Q(x)) \to ((\forall x)P(x) \to (\forall x)Q(x)))$ (1)

— $\alpha$ rule for (1)

$(\forall x)(P(x) \to Q(x))$ (2)

$\neg((\forall x)P(x) \to (\forall x)Q(x))$ (3)

— $\alpha$ rule for (3)

$(\forall x)P(x)$ (4)

$\neg(\forall x)Q(x)$ (5)

— $\delta$ rule for (5), $a$ - a new constant

$\neg Q(a)$

— $\gamma$ rule for (4), $a$ - used for instantiat ion

$P(a)$

$(\forall x)P(x)$ copy of formula (4)

— $\gamma$ rule for (2), $a$ - used for instantiat ion

$P(a) \to Q(a)$ (6)

$(\forall x)(P(x) \to Q(x))$ copy of formula (2)

— $\beta$ rule for (6)

$\neg P(a)$ | $Q(a)$

closed branch $\otimes$ | $\otimes$ closed branch

***$\gamma$ - rules***

$$\begin{array}{cc} (\forall x)A(x) & \neg(\exists x)A(x) \\ | & | \\ A(c_1) & \neg A(c_1) \\ | & | \\ A(c_2) & \neg A(c_2) \\ | & | \\ \dots & \dots \\ | & | \\ A(c_n) & \neg A(c_n) \\ | & | \\ (\forall x)A(x) & \neg(\exists x)A(x) \end{array}$$

***$\delta$ - rules***

$$\begin{array}{cc} (\exists x)A(x) & \neg(\forall x)A(x) \\ | & | \\ A(c) & \neg A(c) \end{array}$$

The semantic tableau of $\neg U_1$ is closed, with two closed branches containing pairs of opposite literals: $(P(a), \neg P(a))$ and $(Q(a), \neg Q(a))$, therefore $\neg U_1$ is inconsistent and $U_1$ is valid.

M.Lupea

The semantic tableau of $\neg U_2$ is ***finite, complete and open***, with one open branch and one closed branch, so $\neg U_2$ ***is consistent*** and $U_2$ ***is not a tautology***.

$\neg U_2 = \neg(((\forall x)P(x) \to (\forall x)Q(x)) \to (\forall x)(P(x) \to Q(x)))$ (1)

| $-\ \alpha$ rule for (1)

$(\forall x)P(x) \to (\forall x)Q(x)$ (2)

|

$\neg(\forall x)(P(x) \to Q(x))$ (3)

| $-\ \delta$ rule for (3), $a$ - new constant

$\neg(P(a) \to Q(a))$ (4)

| $-\ \alpha$ rule for (4)

$P(a)$

|

$\neg Q(a)$

———— $\beta$ rule for (2)

(5) $\neg(\forall x)P(x)$ $\quad$ $(\forall x)Q(x)$ (6)

$\delta$ rule for (5), $b$ - new constant $-$ | $\quad$ | $-\ \gamma$ rule for (6), $a$ - used for instantiat ion

$\neg P(b)$ $\quad$ $Q(a)$

open branch $\odot$

|

$(\forall x)Q(x)$ copy of formula (6)

$\otimes$ closed branch

**Conclusion:** We proved that $\models U_1$ and $\not\models U_2$ so,

***the universal quantifier is not distributive over implication, it is only semi-distributive.***

**Find anti-models of the non-valid formula** $U_2$.

The open branch provides a partial model that covers two complete models of $\neg U_2$: $I_1$ and $I_2$, which are anti-models of $U_2$.

$I_1 = < D, m_1 >$, where $D = \{a, b\}$ – the domain of interpretation

$m_1(P): D \rightarrow \{T, F\}$, $m_1(P)(a) = T$, $m_1(P)(b) = F$,

$m_1(Q): D \rightarrow \{T, F\}$, $m_1(Q)(a) = F$, $m_1(Q)(b) = T$

$v^{I_1}(\neg U_2) = T$ and $v^{I_1}(U_2) = F$.

$I_2 = < D, m_2 >$, where $D = \{a, b\}$ – the domain of interpretation

$m_2(P): D \rightarrow \{T, F\}$, $m_2(P)(a) = T$, $m_2(P)(b) = F$,

$m_2(Q): D \rightarrow \{T, F\}$, $m_2(Q)(a) = F$, $m_2(Q)(b) = F$

$v^{I_2}(\neg U_2) = T$ and $v^{I_2}(U_2) = F$.

M.Lupea

# Example 6: Prove the non-validity of $U = (\exists y)(\forall x)P(x,y)$

$\neg ((\exists y)(\forall x)P(x,y)) \equiv (\forall y)(\exists x)\neg P(x,y)$ (1)

| - $\gamma$ rule for (1), c0- new constant

$(\exists x)\neg P(x,c0)$ (2)

|

$(\forall y)(\exists x)\neg P(x,y)$ (3) copy of formula (1)

| - $\delta$ rule for (2), c1- new constant

$\neg P(c1,c0)$

| - $\gamma$ rule for (3), c1 is used for instantiation

$(\exists x)\neg P(x,c1)$ (4)

|

$(\forall y)(\exists x)\neg P(x,y)$ (5) copy of formula (1)

|- $\delta$ rule for (4), c2- new constant

$\neg P(c2,c1)$

| - $\gamma$ rule for (5), c2 is used for instantiation

$(\exists x)\neg P(x,c2)$ (6)

....

The semantic tableau of $\neg$U is ***infinite* so,**
**we cannot decide upon the validity of U.**

A well known interpretation which falsifies U is
I=<D,m>, where the domain D=N,
the set of all natural numbers and
m(P): NxN ->{T,F}, m(P)(x,y): "x < y".

The formula $(\exists y)(\forall x)P(x,y)$ is evaluated under the interpretation I as:

$$v^I(U) = (\exists y)_{y\in N}(\forall x)_{x\in N}"x < y"$$

and can be translated as:
**"The biggest natural number exists",**
which is obvious false, and thus **U is not a tautology**.