Non-classical logics

Lecture 6: Many-valued logics (3)

Viorica Sofronie-Stokkermans

sofronie@uni-koblenz.de

Until now

- Many-valued logic (finitely-valued; infinitely-valued)

  History and Motivation

  Syntax

  Semantics

Today:

  Functional completeness

  Automated reasoning:

    Tableaux

    Resolution

Functional completeness

Definition A family $(M, \{f_M : M^n \to M\}_{f \in \mathcal{F}})$ is called functionally complete if every function $g : M^m \to M$ can be expressed in terms of the functions $\{f_M : M^n \to M \mid f \in \mathcal{F}\}$.

Definition A many-valued logic with finite set of truth values M and logical operators $\mathcal{F}$ is called functionally complete if for every function $g : M^m \to M$ there exists a propositional formula F of the logic such that for every $\mathcal{A} : \Pi \to M$

$g(\mathcal{A}(x_1), \dots, \mathcal{A}(x_m)) = \mathcal{A}(F)$.

Example: Propositional logic

$F: \quad (P \vee Q) \wedge ((\neg P \wedge Q) \vee R)$

P	Q	R	$(P \vee Q)$	$\neg P$	$(\neg P \wedge Q)$	$((\neg P \wedge Q) \vee R)$	F
0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0
0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1
1	1	0	1	0	0	0	0
1	1	1	1	0	0	1	1

Example: Propositional logic

$F: \quad (P \lor Q) \land ((\neg P \land Q) \lor R)$

P	Q	R	$(P \lor Q)$	$\neg P$	$(\neg P \land Q)$	$((\neg P \land Q) \lor R)$	F
0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0
0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1
1	1	0	1	0	0	0	0
1	1	1	1	0	0	1	1

Example: Propositional logic

$F: \quad (P \vee Q) \wedge ((\neg P \wedge Q) \vee R)$

P	Q	R	$(P \vee Q)$	$\neg P$	$(\neg P \wedge Q)$	$((\neg P \wedge Q) \vee R)$	F
0	0	0	0	1	0	0	0
0	0	1	0	1	0	1	0
0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	0	0	0	0
1	0	1	1	0	0	1	1
1	1	0	1	0	0	0	0
1	1	1	1	0	0	1	1

DNF: $(\neg P \wedge Q \wedge \neg R) \vee (\neg P \wedge Q \wedge R) \vee (P \wedge \neg Q \wedge R) \vee (P \wedge Q \wedge R)$

Functional completeness

Theorem. Propositional logic is functionally complete.

Proof. For every $g : \{0,1\}^m \to \{0,1\}$ let:

$F = \bigvee_{(a_1,\ldots,a_m)\in\{0,1\}} (c_{g(a_1,\ldots,a_m)} \wedge P_1^{a_1} \wedge \cdots \wedge P_m^{a_m})$

where $P^a = \begin{cases} P & \text{if } a = 1 \\ \neg P & \text{if } a = 0 \end{cases}$

(Then clearly $\mathcal{A}(P)^a = 1$ iff $\mathcal{A}(P) = a$, i.e. $1^1 = 0^0 = 1; 1^0 = 0^1 = 0$.)

It can be easily checked that for every $\mathcal{A} : \{P_1, \ldots, P_m\} \to \{0,1\}$ we have:

$g(\mathcal{A}(P_1), \ldots, \mathcal{A}(P_m)) = \mathcal{A}(F)$.

Functional completeness

Theorem. The logic $\mathcal{L}_3$ is not functionally complete.

Proof. If F is a formula with n propositional variables in the language of $\mathcal{L}_3$ with operators $\{\neg, \sim, \vee, \wedge\}$ then for the valuation $\mathcal{A} : \Pi = \{P_1, \dots, P_n\} \rightarrow \{0, u, 1\}$ with $\mathcal{A}(P_i) = 1$ for all i we have: $\mathcal{A}(F) \neq u$.

Therefore: If g is a function which takes value u when the arguments are in $\{0, 1\}$ then there is no formula F such that $g(\mathcal{A}(P_1), \dots, \mathcal{A}(P_n)) = \mathcal{A}(F)$ for all $\mathcal{A} : \Pi \rightarrow \{0, u, 1\}$.

Theorem. $\mathcal{L}_3^+$, obtained from $\mathcal{L}_3$ by adding one more constant operation u (which takes always value u) is functionally complete.

A simple criterion for functional completeness

Theorem. An m-valued logic with set of truth values $M = \{w_1, \dots, w_m\}$ and logical operations $\mathcal{F}$ with truth tables $\{f_M \mid f \in \mathcal{F}\}$ in which the functions:

- $\min(x, y)$, $\max(x, y)$,
- $J_k(x) = \begin{cases} 1 \text{ (maximal element)} & \text{if } k = x \\ 0 \text{ (minimal element)} & \text{otherwise} \end{cases}$
- all constant functions $c^n_k(x_1, \dots, x_n) = k$

can be expressed in terms of the functions $\{f_M \mid f \in \mathcal{F}\}$ is functionally complete.

Proof. Let $g : M^n \to M$.

$$g(x_1, \dots, x_n) = \max\{\min\{c^n_{g(a_1,\dots,a_n)}, J_{a_1}(x_1), \dots, J_{a_n}(x_n)\} \mid (a_1, \dots a_n) \in M^n\}$$

Functional completeness of $\mathcal{L}_3^+$

Theorem. $\mathcal{L}_3^+$, obtained from $\mathcal{L}_3$ by adding one more constant operation u (which takes always value u) is functionally complete.

Proof

- We define $J_1, J_u, J_0 : \{0, u, 1\} \rightarrow \{0, u, 1\}$ as follows:

$J_0(x) = \sim\sim \neg x$

$J_u(x) = \sim x \wedge \sim \neg x$

$J_1(x) = \sim\sim x$

x	$J_0(x)$	$J_u(x)$	$J_1(x)$
0	1	0	0
u	0	1	0
1	0	0	1

Functional completeness of $\mathcal{L}_3^+$

Theorem. $\mathcal{L}_3^+$, obtained from $\mathcal{L}_3$ by adding one more constant operation u (which takes always value u) is functionally complete.

Proof

- We define $J_1, J_u, J_0 : \{0, u, 1\} \to \{0, u, 1\}$ as follows:

$J_0(x) = \sim\sim \neg x$

$J_u(x) = \sim x \wedge \sim \neg x$

$J_1(x) = \sim\sim x$

x	$J_0(x)$	$J_u(x)$	$J_1(x)$
0	1	0	0
u	0	1	0
1	0	0	1

- min and max are $\wedge$ resp. $\vee$.

Functional completeness of $\mathcal{L}_3^+$

Theorem. $\mathcal{L}_3^+$, obtained from $\mathcal{L}_3$ by adding one more constant operation u (which takes always value u) is functionally complete.

Proof

- We define $J_1, J_u, J_0 : \{0, u, 1\} \rightarrow \{0, u, 1\}$ as follows:

$J_0(x) = \sim\sim \neg x$

$J_u(x) = \sim x \wedge \sim \neg x$

$J_1(x) = \sim\sim x$

x	$J_0(x)$	$J_u(x)$	$J_1(x)$
0	1	0	0
u	0	1	0
1	0	0	1

- min and max are $\wedge$ resp. $\vee$.
- The constant operation u is in the language.
- The constant functions 0 and 1 are definable as follows:

$1(x) = \sim x \vee \neg \sim x$
$0(x) = \sim (\sim x \vee \neg \sim x)$

Example

Let g the following binary function:

g	0	u	1
0	0	u	0
u	u	u	u
1	0	u	0

$$\begin{aligned} g(x_1, x_2) = \quad & (u \wedge J_0(x_1) \wedge J_u(x_2)) \vee (u \wedge J_u(x_1) \wedge J_0(x_2)) \vee \\ & (u \wedge J_u(x_1) \wedge J_u(x_2)) \vee (u \wedge J_u(x_1) \wedge J_1(x_2)) \vee (u \wedge J_1(x_1) \wedge J_u(x_2)) \\ = \quad & (u \wedge \sim\sim \neg x_1 \wedge \sim x_2 \wedge \sim \neg x_2) \vee (u \wedge \sim x_1 \wedge \sim \neg x_1 \wedge \sim\sim \neg x_2) \vee \\ & (u \wedge \sim x_1 \wedge \sim \neg x_1 \wedge \sim x_2 \wedge \sim \neg x_2) \vee \\ & (u \wedge \sim x_1 \wedge \sim \neg x_1 \wedge \sim\sim x_2) \vee \\ & (u \wedge \sim\sim x_1 \wedge \sim x_2 \wedge \sim \neg x_2) \end{aligned}$$

Post logics

$P_m = \{0, 1, \ldots, m-1\}$

$\mathcal{F} = \{\vee, s\}$

$\vee_P(a, b) = \max(a, b)$

$s_P(a) = a - 1 \pmod m$

Post logics

Theorem. The Post logics are functionally complete.

Proof:

1. max is $\vee_P$
2. The functions $x - k \pmod{m}$ and $x + k \pmod{m}$ are definable

   $x - k = \underbrace{s(s(...s}_{k \text{ times}}(x))) \pmod{m}$

   $x + k = x - (m - k) \pmod{m}$, $0 < k < m$.

   $x + 0 = x$

3. $\min(x, y) = m - 1 - \max(m - 1 - x, m - 1 - y)$

Post logics

Theorem. The Post logics are functionally complete.

Proof:

4. All constants are definable

$T(x) = max\{x, x-1, \ldots, x-m+1\}$
$T(x) = m-1$ for all x.

The other constants are definable using s iterated $1, 2, \ldots, m-1$ times.

5. $T_k(x) = \max(\max[T(x)-1, x] - m + 1, x + k) - m + 1$ has the property that $T_k(x) = \begin{cases} 0 & \text{if } x \neq m-1 \\ k & \text{if } x = m-1 \end{cases}$

Then $J_k(x) = \max(T_{J_k(0)}(x+m-1), \ldots, T_{J_k(m-2)}(x+1), T_{J_k(m-1)}(x))$.

in general, if $g(i)=k_i$ then $g(x)=\max(T_{k_{m-1}}(x), T_{k_{m-2}}(x+1), \ldots, T_{k_0}(x+(m-1)))$

Proof Calculi and Automated reasoning

- Axiom systems $\mapsto$ proofs
- Tableau calculi
- Resolution calculi

...

Proof Calculi/Inference systems and proofs

Inference systems Γ (proof calculi) are sets of tuples

$$(F_1, \ldots, F_n, F_{n+1}), \;\; n \geq 0,$$

called inferences or inference rules, and written

$$\frac{\overbrace{F_1 \;\; \ldots \;\; F_n}^{\text{premises}}}{\underbrace{F_{n+1}}_{\text{conclusion}}} \; .$$

Inferences with 0 premises are also called axioms.

Clausal inference system: premises and conclusions are clauses. One also considers inference systems over other data structures.

Proofs

A proof in Γ of a formula F from a a set of formulas N (called assumptions) is a sequence $F_1, \ldots, F_k$ of formulas where

(i) $F_k = F$,

(ii) for all $1 \leq i \leq k$: $F_i \in N$, or else there exists an inference $(F_{i_1}, \ \ldots, \ F_{i_{n_i}}, \ F_i)$ in Γ, such that $0 \leq i_j < i$, for $1 \leq j \leq n_i$.

Soundness and Completeness

Provability $\vdash_\Gamma$ of F from N in Γ:

$N \vdash_\Gamma F$:⇔ there exists a proof Γ of F from N.

Γ is called sound :⇔

$$\frac{F_1 \ \ldots \ F_n}{F} \in \Gamma \quad \Rightarrow \quad F_1, \ldots, F_n \models F$$

Γ is called complete :⇔

$$N \models F \quad \Rightarrow \quad N \vdash_\Gamma F$$

Γ is called refutationally complete :⇔

$$N \models \bot \quad \Rightarrow \quad N \vdash_\Gamma \bot$$

Axiom systems

For $\mathcal{L}_3$: Wajsberg proposed an axiom system (based on connectors $\neg$ and $\Rightarrow$):

$A_1 : (A \Rightarrow (B \Rightarrow A))$

$A_2 : (A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C))$

$A_3 : (\neg A \Rightarrow \neg B) \Rightarrow (B \Rightarrow A)$

$A_4 : ((A \Rightarrow \neg A) \Rightarrow A) \Rightarrow A$

Inference rules:

Moduls Ponens: $$\frac{A \qquad A \Rightarrow B}{B}$$

Axiom systems

For $\mathcal{L}_3$: Wajsberg proposed an axiom system (based on connectors $\neg$ and $\Rightarrow$):

$x \wedge y = x \cdot (x \Rightarrow y)$,

where $x \cdot y = \neg(x \Rightarrow \neg y)$

Proof calculi

Main disadvantage:

New proof calculus for each many-valued logic.

Goal:

Uniform methods for checking validity/satisfiability of formulae.

Automated reasoning

Classical logic:

Task: prove that F is valid

Method: prove that $\neg F$ is unsatisfiable:

– assume $\neg F$; derive a contradiction.

Automated reasoning

Classical logic:

Task: prove that F is valid

Method: prove that $\neg F$ is unsatisfiable:

– assume $\neg F$; derive a contradiction.

Many-valued logic:

Task: prove that F is valid

(i.e. $\mathcal{A}(\beta)(F) \in D$ for all $\mathcal{A}, \beta$)

Method: prove that it is not possible that $\mathcal{A}(\beta) \in M \backslash D$:

– assume $F \in M \backslash D$; derive a contradiction.

Automated reasoning

Classical logic:

Task: prove that F is valid

Method: prove that $\neg F$ is unsatisfiable:

– assume $\neg F$; derive a contradiction.

Many-valued logic:

Task: prove that F is valid

(i.e. $\mathcal{A}(\beta)(F) \in D$ for all $\mathcal{A}, \beta$)

Method: prove that it is not possible that $\mathcal{A}(\beta) \in M\backslash D$:

– assume $F \in M\backslash D$; derive a contradiction.

Problem: How do we express the fact that $F \in M\backslash D$

1) $\bigvee_{v \in M\backslash D}(F = v)$

2) more economical notation?

Automated reasoning

Idea: Use signed formulae

- F^v, where F is a formula and $v \in M$

  $\mathcal{A}, \beta \models F^v$ iff $\mathcal{A}(\beta)(F) = v$.

- $S{:}F$, where F is a formula and
  $\emptyset \neq S \subseteq M$ (set of truth values)

  $\mathcal{A}, \beta \models S{:}F$ iff $\mathcal{A}(\beta)(F) \in S$.

Semantic tableaux

For every $\emptyset \neq S \subseteq M$ and every logical operator f we have a tableau rule:

$$\frac{S{:}f(F_1, \ldots, F_n)}{T(F_1, \ldots, F_n)}$$

where $T(A_1, \ldots, A_n)$ is a finite extended tableau containing only formulae of the form $S_i{:}F_i$.

Informally: Exhaustive list of conditions which ensure that the value of $f(F_1, \ldots, F_n)$ is in S.

Example

Let $Ł_5$ be the 5-valued Łukasiewicz logic with $M = \{0, 1, 2, 3, 4\}$.

$\Rightarrow$	0	1	2	3	4
0	4	4	4	4	4
1	3	4	4	4	4
2	2	3	4	4	4
3	1	2	3	4	4
4	0	1	2	3	4

$$\{4\}(p \Rightarrow q)$$

$\{0\}p$	$\{0, 1\}p$	$\{0, 1, 2\}p$	$\{0, 1, 2, 3\}p$	
	$\{1, 2, 3, 4\}q$	$\{2, 3, 4\}q$	$\{3, 4\}q$	$\{4\}q$

Labelling sets

Let $V \subseteq \mathcal{P}(M)$ be the set of all sets of truth values which are used for labelling the formulae.

Remarks:

1. In general not all subsets of truth values are used, so $V \neq \mathcal{P}(M)$.

2. Proof by contradiction:
   Goal: Prove that F is valid, i.e. $\mathcal{A}(\beta)(F) \in D$.
   We start from $(M \backslash D)$:F and build the tableau
   $\Rightarrow$ We assume that $(M \backslash D) \in V$.

3. Need to make sure that the new signs introduced by the tableau rules are in V.

Tableau rules: Soundness

$$\frac{S{:}f(F_1, \ldots, F_n)}{T(F_1, \ldots, F_n)}$$

where $T(F_1, \ldots, F_n)$ is a finite extended tableau containing only formulae of the form $S_i{:}F_i$.

$$S{:}f(F_1, \ldots, F_n)$$

$S_{11}{:}C_{11}$	$S_{21}{:}C_{21}$	$\ldots$	$S_{q1}{:}C_{q1}$
$\ldots$	$\ldots$		$\ldots$
$S_{1k_1}{:}C_{1k_1}$	$S_{2k_2}{:}C_{2k_2}$		$S_{qk'}{:}C_{qk'}$

where $C_{i,j} \in \{F_1, \ldots, F_n\}$

Tableau rules: Soundness

$$\frac{S{:}f(F_1, \ldots, F_n)}{T(F_1, \ldots, F_n)}$$

where $T(F_1, \ldots, F_n)$ is a finite extended tableau containing only formulae of the form $S_i{:}F_i$.

$$\begin{array}{c|c|c|c} \multicolumn{4}{c}{S{:}f(F_1, \ldots, F_n)} \\ \hline S_{11}{:}C_{11} & S_{21}{:}C_{21} & \ldots & S_{q1}{:}C_{q1} \\ \ldots & \ldots & & \ldots \\ S_{1k_1}{:}C_{1k_1} & S_{2k_2}{:}C_{2k_2} & & S_{qk'}{:}C_{qk'} \end{array}$$

where $C_{i,j} \in \{F_1, \ldots, F_n\}$

For every $\mathcal{A}, \beta$: $\mathcal{A}(\beta)(F) \in S$ then there exists i such that for all j: $\mathcal{A}(\beta)(C_{ij}) \in S_{ij}$.

Tableau rules: Soundness

$$\frac{S{:}f(F_1, \ldots, F_n)}{\begin{array}{c|c|c|c} S_{11}{:}C_{11} & S_{21}{:}C_{21} & \ldots & S_{q1}{:}C_{q1} \\ \ldots & \ldots & & \ldots \\ S_{1k_1}{:}C_{1k_1} & S_{2k_2}{:}C_{2k_2} & & S_{qk'}{:}C_{qk'} \end{array}}$$

where $C_{i,j} \in \{F_1, \ldots, F_n\}$

Every model of $S{:}f(F_1, \ldots, F_n)$ is also a model of the formulae on one of the branches

If there is no expansion rule for a premise: premise is unsatisfiable $(\mathcal{A}(\beta)(F) \notin S$ for all $\mathcal{A}, \beta)$.

If $f(F_1, \ldots, F_n)$ satisfiable then there is an expansion rule.

$\mathcal{L}_3$: Tableau rules for $\wedge$

$$\frac{\{1\}A \wedge B}{\begin{array}{c}\{1\}A \\ \{1\}B\end{array}} \qquad \frac{\{u\}A \wedge B}{\begin{array}{c|c|c}\{u\}A & \{u\}B & \{u\}A \\ \{1\}B & \{1\}A & \{u\}B\end{array}} \qquad \frac{\{0\}A \wedge B}{\{0\}A \mid \{0\}B} \qquad \frac{\{u,0\}A \wedge B}{\{u,0\}A \mid \{u,0\}B}$$

$\mathcal{L}_3$: Tableau rules for $\vee$

$$\frac{\{1\}A \vee B}{\{1\}A \mid \{1\}B}$$

$$\frac{\{u\}A \vee B}{\begin{array}{c|c} \{u,0\}A & \{u\}A \\ \{u\}B & \{u,0\}B \end{array}}$$

$$\frac{\{0\}A \vee B}{\begin{array}{c} \{0\}A \\ \{0\}B \end{array}}$$

$$\frac{\{u,0\}A \vee B}{\begin{array}{c} \{u,0\}A \\ \{u,0\}B \end{array}}$$

$\mathcal{L}_3$: Tableau rules for $\neg, \sim$

$$\frac{\{1\} \sim A}{\{u,0\}A} \qquad \frac{\{0\} \sim A}{\{1\}A} \qquad \frac{\{u\} \sim A}{} \qquad \frac{\{u,0\} \sim A}{\{1\}A}$$

$$\frac{\{1\}\neg A}{\{0\}A} \qquad \frac{\{0\}\neg A}{\{1\}A} \qquad \frac{\{u\}\neg A}{\{u\}A} \qquad \frac{\{u,0\}\neg A}{\{1\}A|\{u\}A}$$

$\mathcal{L}_3$: Tableau rules for $\supset$

$$\frac{\{1\}A \supset B}{\{u,0\}A \mid \{1\}B} \qquad \frac{\{0\}A \supset B}{\begin{array}{c}\{1\}A\\ \{0\}B\end{array}} \qquad \frac{\{u\}A \supset B}{\begin{array}{c}\{1\}A\\ \{u\}B\end{array}} \qquad \frac{\{u,0\}A \supset B}{\begin{array}{c}\{1\}A\\ \{u,0\}B\end{array}}$$

$\mathcal{L}_3$: Tableau rules for $\exists$

$$\frac{\{1\}\exists x A(x)}{\{1\}A(f(y_1,\ldots,y_k))} \qquad \frac{\{0\}\exists x A(x)}{\{0\}A(z)} \qquad \frac{\{u\}\exists x A(x)}{\begin{array}{c}\{u\}A(f(y_1,\ldots y_k)) \\ \{u,0\}A(z)\end{array}} \qquad \frac{\{u,0\}\exists x A(x)}{\{u,0\}A(z)}$$

where

- z is a new free variable
- $y_1,\ldots,y_k$ are the free variables in $\exists x A(x)$
- f is a new function symbol

$\mathcal{L}_3$: Tableau rules for $\forall$

$$\frac{\{1\}\forall x A(x)}{\{1\}A(z)} \qquad \frac{\{0\}\forall x A(x)}{\{0\}A(f(y_1,\ldots,y_k))} \qquad \frac{\{u\}\forall x A(x)}{\begin{array}{c}\{u\}A(f(y_1,\ldots y_k)) \\ \{u,1\}A(z)\end{array}} \qquad \frac{\{u,0\}\forall x A(x)}{\{u,0\}A(f(y_1,\ldots,y_k))}$$

where

- z is a new free variable
- $y_1,\ldots,y_k$ are the free variables in $\forall x A(x)$
- f is a new function symbol

Tableaux

A tableau for a finite set For of signed formulae is constructed as follows:

- A linear tree, in which each formula in For occurs once is a tableau.
- Let T be a tableau for For und P a path in T, which contains a signed formula $S{:}F$.

  Assume that there exists a tableau rule with premise $S{:}F$. If $E_1, \ldots, E_n$ are the possible conclusions of the tableau rule (under the corresponding restrictions in case of quantified formulae) then T is exteded with n linear subtrees containing the signed formulae from E_i (respectively), in arbitrary order.

  The tree obtained this way is again a tableau for For.

Closed Tableaux

A path P in a tableau T is closed if:

- P contains complementary formulae, i.e. there exists a substitution σ and there exists signed formulae $S_1{:}F_1, \ldots, S_k{:}F_k$ occurring of the branch such that:
  - $F_1\sigma = \cdots = F_n\sigma$
  - $S_1 \cap \cdots \cap S_n = \emptyset$, or
- P contains a signed formula $S{:}F$ for which no expansion rule can be applied and F is not atomic.

A path which is not closed is called open.

Closed Tableaux

A path P in a tableau T is closed if:

- P contains complementary formulae, i.e. there exists a substitution σ and there exists signed formulae $S_1{:}F_1, \ldots, S_k{:}F_k$ occurring of the branch such that:
  - $F_1\sigma = \cdots = F_n\sigma$
  - $S_1 \cap \cdots \cap S_n = \emptyset$, or
- P contains a signed formula $S{:}F$ for which no expansion rule can be applied and F is not atomic.

A path which is not closed is called open.

A tableau is closed if every path can be closed with the same substitution.

Otherwise the tableau is called open.

Soundness and completeness

Given an signature Σ, by Σ^{sko} we denote the result of adding infinitely many new Skolem function symbols which we may use in the rules for quantifiers.

Let $\mathcal{A}$ be a Σ^{sko}-interpretation, T a tableau, and β a variable assignment over $\mathcal{A}$.

T is called $(\mathcal{A}, \beta)$-valid, if there is a path P_β in T such that $\mathcal{A}, \beta \models F$, for each formula F on P_β.

T is called satisfiable if there exists a structure $\mathcal{A}$ such that for each assignment β the tableau T is $(\mathcal{A}, \beta)$-valid.
(This implies that we may choose P_β depending on β.)

Soundness and completeness

Theorem (Soundness of the tableau calculus for $\mathcal{L}_3$)
Let F be a $\mathcal{L}_3$-formula without free variables. If there exists a closed tableau T for $\{U, F\}F$, then F is an $\mathcal{L}_3$-tautology (it is valid).

Theorem (Refutational completeness)
Let F be a $\mathcal{L}_3$-tautology. Then we can construct a closed tableau for $\{U, F\}F$. (The order in which we apply the expansion rules is not important).