DTU Course 02156 Logical Systems and Logic Programming (2021)

Week	Date	Main Topics (Prolog Programming in All Lessons)
35 #01	31/8	Course Prerequisites & Tutorial on Logical Systems and Logic Programming
36 #02	7/9	Chapter 1 - Introduction (Prolog Note)
37 #03	14/9	Chapter 2 - Propositional Logic: Formulas, Models, Tableaux
38 #04	21/9	Chapter 3 - Propositional Logic: Deductive Systems
39 #05	28/9	"Isabelle" - Propositional Logic: Sequent Calculus Verifier (SeCaV)
40 #06	5/10	Chapter 4 - Propositional Logic: Resolution
41 #07	12/10	Chapter 7 - First-Order Logic: Formulas, Models, Tableaux
42		(Autumn Vacation)
43 #08	26/10	Chapter 8 - First-Order Logic: Deductive Systems
44 #09	2/11	"Isabelle" - First-Order Logic: Sequent Calculus Verifier (SeCaV)
45 #10	9/11	Chapter 9 - First-Order Logic: Terms and Normal Forms
46 #11	16/11	Chapter 10 - First-Order Logic: Resolution
47 #12	23/11	Chapter 11 - First-Order Logic: Logic Programming
48 #13	30/11	Chapter 12 - First-Order Logic: Undecidability and Model Theory & Course Evaluation

Responsible: Associate Professor Jørgen Villadsen <jovi@dtu.dk>

Assignments & Exam MUST BE SOLVED INDIVIDUALLY

Assignment-1 Deadline Sunday 26/9 (Available Wednesday 15/9)

Assignment-2 Deadline Sunday 10/10 (Available Wednesday 29/9)

Assignment-3 Deadline Sunday 31/10 (Available Wednesday 13/10)

Assignment-4 Deadline Sunday 14/11 (Available Wednesday 3/11)

Assignment-5 Deadline Thursday 2/12 (Available Wednesday 17/11)

Written Exam Tuesday 14/12 (2 Hours / No Computer / All Notes Allowed)

The mandatory assignments and the written exam are evaluated as a whole – even if you do well in the mandatory assignments then you still must do decent in the written exam in order to pass the course!

A TEACHER MUST IMMEDIATELY REPORT ANY SUSPICION OF CHEATING TO THE STUDY ADMINISTRATION FOR FURTHER ACTIONS

THE KEPLER CONJECTURE

The Kepler conjecture, named after the 17th-century mathematician and astronomer Johannes Kepler, is a mathematical theorem about sphere packing in three-dimensional Euclidean space. It says that no arrangement of equally sized spheres filling space has a greater average density than that of the cubic close packing (face-centered cubic) and hexagonal close packing arrangements. The density of these arrangements is around 74.05%.

In 1998 Thomas Hales, following an approach suggested by Fejes Tóth (1953), announced that he had a proof of the Kepler conjecture. Hales' proof is a proof by exhaustion involving the checking of many individual cases using complex computer calculations. Referees said that they were "99% certain" of the correctness of Hales' proof, and the Kepler conjecture was accepted as a theorem. In 2014, the Flyspeck project team, headed by Hales, announced the completion of a formal proof of the Kepler conjecture using a combination of the Isabelle and HOL Light proof assistants. In 2017, the formal proof was accepted into the Forum of Mathematics journal.

https://en.wikipedia.org/wiki/Kepler_conjecture

METAMATH

Metamath is a language for developing strictly formalized mathematical definitions and proofs accompanied by a proof checker for this language and a growing database of thousands of proved theorems covering conventional results in logic, set theory, number theory, group theory, algebra, analysis, and topology…

https://en.wikipedia.org/wiki/Metamath

Agenda — Week #7

How are you doing?

Tableaux & Resolution...

First-Order Logic (FOL)

How are you doing?

Write a Prolog program `same_length(+List1,+List2)` that succeeds if and only if the two lists have the same number of elements.

Write two variants: one using only the predicate `length` and another not using any other predicate except possibly `same_length` (hence it can be recursive).

Sample queries:

```
?- same_length([],[]).

Yes
```

```
?- same_length([],[a]).

No
```

```
?- same_length([a,b],[b,a]).

Yes
```

```
?- same_length([a,b],[a]).

No
```

Tableaux. . .

You must master it for propositional logic by now!

```
~((p => (q => r)) => ((~q => r) => ((~p => r) => r)))
   (p => (q => r)),~((~q => r) => ((~p => r) => r))
      (~q => r),~((~p => r) => r),(p => (q => r))
         (~p => r),~r,(~q => r),(p => (q => r))
            ~~p,~r,(~q => r),(p => (q => r))
               p,~r,(~q => r),(p => (q => r))
                  ~~q,p,~r,(p => (q => r))
                     q,p,~r,(p => (q => r))
                        ~p,q,p,~r Closed
                        (q => r),q,p,~r
                           ~q,q,p,~r Closed
                           r,q,p,~r Closed
                  r,p,~r,(p => (q => r)) Closed
            r,~r,(~q => r),(p => (q => r)) Closed
```

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Another set of axioms:

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Another set of axioms:

$\vdash A \vee A \rightarrow A$

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Another set of axioms:

$\vdash A \vee A \rightarrow A$

$\vdash A \rightarrow A \vee B$

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Another set of axioms:

$\vdash A \lor A \to A$

$\vdash A \to A \lor B$

$\vdash A \lor B \to B \lor A$

Deductive Systems

In the textbook it is shown how to transform a closed tableau for the negation of a formula into a proof in the Gentzen system — and any proof in the Gentzen system into a proof in the Hilbert system

Another set of axioms:

$\vdash A \vee A \rightarrow A$

$\vdash A \rightarrow A \vee B$

$\vdash A \vee B \rightarrow B \vee A$

$\vdash (B \rightarrow C) \rightarrow (A \vee B \rightarrow A \vee C)$

CNF / Resolution — Last Week — Exercise 1

```
(p => q => r) => (~q => r) => (~p => r) => r
```

```
~((p => (q => r)) => ((~q => r) => ((~p => r) => r)))
~(~(~p \ (~q \ r)) \ (~(~~q \ r) \ (~(~~p \ r) \ r)))
((~p \ (~q \ r)) & ((q \ r) & ((p \ r) & ~r)))
```

1.	$\overline{p}\,\overline{q}\,r$	
2.	$p\,r$	
3.	$q\,r$	
4.	$\overline{r}$	
5.	$\overline{q}\,r$	1,2
6.	r	5,3
7.	$\square$	6,4

Resolution...

```
?- resolution(p).
~p
~p
[[neg p]]

No
```

```
?- resolution(p<=>p).
~ (p<=>p)
(~p\ ~p)& (p\p)
[[neg p],[p]]
[[],[neg p],[p]]

Yes
```

...

```
?- resolution(p<=>p<=>p<=>p<=>p<=>p<=>p<=>p).
~ (p<=>p<=>p<=>p<=>p<=>p<=>p)
ERROR: Out of global stack
```

Much more interesting for First-Order Logic...

Hilbert's Thesis

"Many logicians would contend that there is no logic beyond first-order logic, in the sense that when one is forced to make all one's mathematical (extra-logical) assumptions explicit, these axioms can always be expressed in first-order logic...

... we refer to this view as *Hilbert's Thesis*."

Jon Barwise (editor): Handbook of Mathematical Logic - Page 41

`books.google.dk/books?isbn=0444863885`

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

What is the longest path for the theorem $2 + 2 = 4$?

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

What is the longest path for the theorem $2 + 2 = 4$?

A longest path back to an axiom from $2 + 2 = 4$ is 150...

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

What is the longest path for the theorem $2 + 2 = 4$?

A longest path back to an axiom from $2 + 2 = 4$ is 150...

By the way, the complete proof of $2 + 2 = 4$ involves 2,452 subtheorems including the 150 above...

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

What is the longest path for the theorem $2 + 2 = 4$?

A longest path back to an axiom from $2 + 2 = 4$ is 150...

By the way, the complete proof of $2 + 2 = 4$ involves 2,452 subtheorems including the 150 above...

These have a total of 25,933 steps — this is how many steps you would have to examine if you wanted to verify the proof by hand in complete detail all the way back to the axioms.

A Formal Proof of $2 + 2 = 4$ in Metamath

The complete proof of a theorem all the way back to axioms can be thought of as a tree of subtheorems, with the steps in each proof branching back to earlier subtheorems, until axioms are ultimately reached at the tips of the branches.

An interesting exercise is, starting from a theorem, to try to find the longest path back to an axiom...

What is the longest path for the theorem $2 + 2 = 4$?

A longest path back to an axiom from $2 + 2 = 4$ is 150...

By the way, the complete proof of $2 + 2 = 4$ involves 2,452 subtheorems including the 150 above...

These have a total of 25,933 steps — this is how many steps you would have to examine if you wanted to verify the proof by hand in complete detail all the way back to the axioms.

One of the reasons that the proof of $2 + 2 = 4$ is so long is that 2 and 4 are complex numbers...

Proof Systems

The textbook describes 3 variants of formal proofs:

- “Gentzen System” — Tableaux
- “Hilbert System” — Axiomatics
- Resolution

They yield the same theorems, that is, formulas with formal proofs, $\{A \mid\ \vdash A\}$, although the proof systems are very different.

Proof Systems

The textbook describes 3 variants of formal proofs:

- "Gentzen System" — Tableaux
- "Hilbert System" — Axiomatics
- Resolution

They yield the same theorems, that is, formulas with formal proofs, $\{A \mid\, \vdash A\}$, although the proof systems are very different.

A Prolog implementation of $\vdash$ based on resolution will be provided.

Proof Systems

The textbook describes 3 variants of formal proofs:

- "Gentzen System" — Tableaux
- "Hilbert System" — Axiomatics
- Resolution

They yield the same theorems, that is, formulas with formal proofs, $\{A \mid \vdash A\}$, although the proof systems are very different.

A Prolog implementation of $\vdash$ based on resolution will be provided.

Call the program `qed` (QED is an abbreviation of the Latin phrase "quod erat demonstrandum" — literally "which was to be proved" — and used to mark the end of a proof).

Proof Systems

The textbook describes 3 variants of formal proofs:

- "Gentzen System" — Tableaux
- "Hilbert System" — Axiomatics
- Resolution

They yield the same theorems, that is, formulas with formal proofs, $\{A \mid \vdash A\}$, although the proof systems are very different.

A Prolog implementation of $\vdash$ based on resolution will be provided.

Call the program `qed` (QED is an abbreviation of the Latin phrase "quod erat demonstrandum" — literally "which was to be proved" — and used to mark the end of a proof).

For example $\vdash (\exists x \forall y p(x,y)) \rightarrow \forall y \exists x p(x,y)$ using `qed`.

Proof Systems

The textbook describes 3 variants of formal proofs:

- "Gentzen System" — Tableaux
- "Hilbert System" — Axiomatics
- Resolution

They yield the same theorems, that is, formulas with formal proofs, $\{A \mid \vdash A\}$, although the proof systems are very different.

A Prolog implementation of $\vdash$ based on resolution will be provided.

Call the program `qed` (QED is an abbreviation of the Latin phrase "quod erat demonstrandum" — literally "which was to be proved" — and used to mark the end of a proof).

For example $\vdash (\exists x \forall y p(x, y)) \rightarrow \forall y \exists x p(x, y)$ using `qed`.

Hence in principle `qed` can produce a formal proof of the famous Kepler Conjecture — in 2014 it was proved using a combination of the Isabelle and HOL Light proof assistants.

Predicate Formulas

$p(x, y)$ has two free variables x and y.

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

$\forall x \forall y p(x, y)$ is the universal closure of $p(x, y)$.

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

$\forall x \forall y p(x, y)$ is the universal closure of $p(x, y)$.

$\exists x \exists y p(x, y)$ is the existential closure of $p(x, y)$.

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

$\forall x \forall y p(x, y)$ is the universal closure of $p(x, y)$.

$\exists x \exists y p(x, y)$ is the existential closure of $p(x, y)$.

The quantifiers are duals:

$$\forall x A(x) \leftrightarrow \neg \exists x \neg A(x) \qquad \exists x A(x) \leftrightarrow \neg \forall x \neg A(x)$$

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

$\forall x \forall y p(x, y)$ is the universal closure of $p(x, y)$.

$\exists x \exists y p(x, y)$ is the existential closure of $p(x, y)$.

The quantifiers are duals:

$$\forall x A(x) \leftrightarrow \neg \exists x \neg A(x) \qquad \exists x A(x) \leftrightarrow \neg \forall x \neg A(x)$$

A few valid formulas:

$$(\forall x A(x) \lor \forall x B(x)) \to \forall x (A(x) \lor B(x))$$
$$\exists x (A(x) \land B(x)) \to (\exists x A(x) \land \exists x B(x))$$

Predicate Formulas

$p(x, y)$ has two free variables x and y.

$\exists y p(x, y)$ has one free variable x (the variable y is bound).

$\forall x \exists y p(x, y)$ is closed (has no free variables).

$\forall x \forall y p(x, y)$ is the universal closure of $p(x, y)$.

$\exists x \exists y p(x, y)$ is the existential closure of $p(x, y)$.

The quantifiers are duals:

$$\forall x A(x) \leftrightarrow \neg \exists x \neg A(x) \qquad \exists x A(x) \leftrightarrow \neg \forall x \neg A(x)$$

A few valid formulas:

$$(\forall x A(x) \vee \forall x B(x)) \rightarrow \forall x (A(x) \vee B(x))$$
$$\exists x (A(x) \wedge B(x)) \rightarrow (\exists x A(x) \wedge \exists x B(x))$$

The implications cannot be reversed.

Interpretations & Assignments

Let U be a set of formulas such that $\{p_1, \ldots, p_m\}$ are all the predicates and $\{a_1, \ldots, a_k\}$ are all the constants appearing in U.

Interpretations & Assignments

Let U be a set of formulas such that $\{p_1, \ldots, p_m\}$ are all the predicates and $\{a_1, \ldots, a_k\}$ are all the constants appearing in U.

An *interpretation* $\mathcal{I}$ is a triple

$$(D, \{R_1, \ldots, R_m\}, \{d_1, \ldots, d_k\})$$

where D is a *non-empty* domain, R_i is an assignment of an n_i-ary relation on D to the n_i-ary predicate p_i, and d_i is an assignment of an element of D to the constant a_i.

Interpretations & Assignments

Let U be a set of formulas such that $\{p_1, \ldots, p_m\}$ are all the predicates and $\{a_1, \ldots, a_k\}$ are all the constants appearing in U.

An *interpretation* $\mathcal{I}$ is a triple

$$(D, \{R_1, \ldots, R_m\}, \{d_1, \ldots, d_k\})$$

where D is a *non-empty* domain, R_i is an assignment of an n_i-ary relation on D to the n_i-ary predicate p_i, and d_i is an assignment of an element of D to the constant a_i.

The formula $\forall x p(a, x)$ has an interpretation $(\mathcal{N}, \{\leq\}, \{0\})$.

Interpretations & Assignments

Let U be a set of formulas such that $\{p_1, \ldots, p_m\}$ are all the predicates and $\{a_1, \ldots, a_k\}$ are all the constants appearing in U.

An *interpretation* $\mathcal{I}$ is a triple

$$(D, \{R_1, \ldots, R_m\}, \{d_1, \ldots, d_k\})$$

where D is a *non-empty* domain, R_i is an assignment of an n_i-ary relation on D to the n_i-ary predicate p_i, and d_i is an assignment of an element of D to the constant a_i.

The formula $\forall x p(a, x)$ has an interpretation $(\mathcal{N}, \{\leq\}, \{0\})$.

Given an interpretation $\mathcal{I}$ an *assignment* $\sigma_{\mathcal{I}}$ is a function which maps every variable to an element of the domain of $\mathcal{I}$.

Interpretations & Assignments

Let U be a set of formulas such that $\{p_1, \ldots, p_m\}$ are all the predicates and $\{a_1, \ldots, a_k\}$ are all the constants appearing in U.

An *interpretation* $\mathcal{I}$ is a triple

$$(D, \{R_1, \ldots, R_m\}, \{d_1, \ldots, d_k\})$$

where D is a *non-empty* domain, R_i is an assignment of an n_i-ary relation on D to the n_i-ary predicate p_i, and d_i is an assignment of an element of D to the constant a_i.

The formula $\forall x p(a, x)$ has an interpretation $(\mathcal{N}, \{\leq\}, \{0\})$.

Given an interpretation $\mathcal{I}$ an *assignment* $\sigma_{\mathcal{I}}$ is a function which maps every variable to an element of the domain of $\mathcal{I}$.

$\sigma_{\mathcal{I}}[x_i \leftarrow d_i]$ is an assignment that is the same as $\sigma_{\mathcal{I}}$ except that the variable x_i is mapped to d_i.

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Let $A = p_k(c_1, \dots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_{\mathcal{I}}}(A) = T$ iff $(d_1, \dots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_{\mathcal{I}}$

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Let $A = p_k(c_1, \ldots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_{\mathcal{I}}}(A) = T$ iff $(d_1, \ldots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_{\mathcal{I}}$

$v_{\sigma_{\mathcal{I}}}(\neg A_1) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = F$

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Let $A = p_k(c_1, \ldots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_{\mathcal{I}}}(A) = T$ iff $(d_1, \ldots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_{\mathcal{I}}$

$v_{\sigma_{\mathcal{I}}}(\neg A_1) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = F$

$v_{\sigma_{\mathcal{I}}}(A_1 \wedge A_2) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = T$ and $v_{\sigma_{\mathcal{I}}}(A_2) = T$

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Let $A = p_k(c_1, \ldots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_{\mathcal{I}}}(A) = T$ iff $(d_1, \ldots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_{\mathcal{I}}$

$v_{\sigma_{\mathcal{I}}}(\neg A_1) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = F$

$v_{\sigma_{\mathcal{I}}}(A_1 \wedge A_2) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = T$ and $v_{\sigma_{\mathcal{I}}}(A_2) = T$

$v_{\sigma_{\mathcal{I}}}(A_1 \vee A_2) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = T$ or $v_{\sigma_{\mathcal{I}}}(A_2) = T$
...

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_{\mathcal{I}}$ an assignment.

$v_{\sigma_{\mathcal{I}}}$, the *value* of A under $\sigma_{\mathcal{I}}$, is defined as follows:

Let $A = p_k(c_1, \ldots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_{\mathcal{I}}}(A) = T$ iff $(d_1, \ldots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_{\mathcal{I}}$

$v_{\sigma_{\mathcal{I}}}(\neg A_1) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = F$

$v_{\sigma_{\mathcal{I}}}(A_1 \wedge A_2) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = T$ and $v_{\sigma_{\mathcal{I}}}(A_2) = T$

$v_{\sigma_{\mathcal{I}}}(A_1 \vee A_2) = T$ iff $v_{\sigma_{\mathcal{I}}}(A_1) = T$ or $v_{\sigma_{\mathcal{I}}}(A_2) = T$
...

$v_{\sigma_{\mathcal{I}}}(\forall x A_1) = T$ iff $v_{\sigma_{\mathcal{I}}[x \leftarrow d]}(A_1) = T$ for *all* $d \in D$

Valuation

Let A be a formula, $\mathcal{I}$ an interpretation and $\sigma_\mathcal{I}$ an assignment.

$v_{\sigma_\mathcal{I}}$, the *value* of A under $\sigma_\mathcal{I}$, is defined as follows:

Let $A = p_k(c_1, \ldots c_n)$ be an atomic formula where each c_i is either a variable x_i or a constant a_i
$v_{\sigma_\mathcal{I}}(A) = T$ iff $(d_1, \ldots, d_n) \in R_k$
where R_k is the relation assigned by $\mathcal{I}$ to p_k, and d_i is the domain element assigned to c_i either by $\mathcal{I}$ or by $\sigma_\mathcal{I}$

$v_{\sigma_\mathcal{I}}(\neg A_1) = T$ iff $v_{\sigma_\mathcal{I}}(A_1) = F$

$v_{\sigma_\mathcal{I}}(A_1 \wedge A_2) = T$ iff $v_{\sigma_\mathcal{I}}(A_1) = T$ and $v_{\sigma_\mathcal{I}}(A_2) = T$

$v_{\sigma_\mathcal{I}}(A_1 \vee A_2) = T$ iff $v_{\sigma_\mathcal{I}}(A_1) = T$ or $v_{\sigma_\mathcal{I}}(A_2) = T$
...

$v_{\sigma_\mathcal{I}}(\forall x A_1) = T$ iff $v_{\sigma_\mathcal{I}[x \leftarrow d]}(A_1) = T$ for *all* $d \in D$

$v_{\sigma_\mathcal{I}}(\exists x A_1) = T$ iff $v_{\sigma_\mathcal{I}[x \leftarrow d]}(A_1) = T$ for *some* $d \in D$

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

A is *unsatisfiable* iff it is not satisfiable.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

A is *unsatisfiable* iff it is not satisfiable.

A is *falsifiable* iff it is not valid.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

A is *unsatisfiable* iff it is not satisfiable.

A is *falsifiable* iff it is not valid.

Gentzen and Hilbert systems $\vdash A$ to be provided.

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

A is *unsatisfiable* iff it is not satisfiable.

A is *falsifiable* iff it is not valid.

Gentzen and Hilbert systems $\vdash A$ to be provided.

Soundness and Completeness Theorem (1930): $\vDash A$ **iff** $\vdash A$

Satisfiability & Validity

If A is closed then $v_{\sigma_{\mathcal{I}}}$ does not depend on $\sigma_{\mathcal{I}}$ (only on $\mathcal{I}$).

Hence for a closed formula A the value is denoted $v_{\mathcal{I}}(A)$.

A closed formula A is *satisfiable* iff $v_{\mathcal{I}}(A) = T$ for *some* interpretation $\mathcal{I}$.

A satisfying interpretation is called a *model* for A, denoted $\mathcal{I} \vDash A$.

A closed formula A is *valid*, denoted $\vDash A$, iff $v_{\mathcal{I}}(A) = T$ for *all* interpretations $\mathcal{I}$.

A is *unsatisfiable* iff it is not satisfiable.

A is *falsifiable* iff it is not valid.

Gentzen and Hilbert systems $\vdash A$ to be provided.

Soundness and Completeness Theorem (1930): $\vDash A$ iff $\vdash A$

Proved by Gödel and later improved by others.

Main Rules for Tableaux — Alpha & Beta — Summary

α	α_1	α_2
$\neg\neg A_1$	A_1	
$A_1 \wedge A_2$	A_1	A_2
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$
$\neg(A_1 \rightarrow A_2)$	A_1	$\neg A_2$
$A_1 \leftrightarrow A_2$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$
$\neg(A_1 \oplus A_2)$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$

β	β_1	β_2
$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$B_1 \vee B_2$	B_1	B_2
$B_1 \rightarrow B_2$	$\neg B_1$	B_2
$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$
$B_1 \oplus B_2$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$

Main Rules for Tableaux — Gamma & Delta — Summary

γ	$\gamma(a)$
$\forall x A(x)$	$A(a)$
$\neg\exists x A(x)$	$\neg A(a)$

Only use if no α-, β- or δ-formula

Use all constants and all γ-formulas — and keep the γ-formulas

Just add a new constant a if there are no constants

δ	$\delta(a)$
$\exists x A(x)$	$A(a)$
$\neg\forall x A(x)$	$\neg A(a)$

Constant a must be new in δ-formula

Tableaux — A Systematic Search for a Counterexample

$$(\exists x p(x) \rightarrow \exists x q(x)) \rightarrow \forall x (p(x) \rightarrow q(x))$$

Tableaux — A Systematic Search for a Counterexample

$$(\exists x p(x) \rightarrow \exists x q(x)) \rightarrow \forall x(p(x) \rightarrow q(x))$$

$\neg((\exists x p(x) \rightarrow \exists x q(x)) \rightarrow \forall x(p(x) \rightarrow q(x)))$
$\exists x p(x) \rightarrow \exists x q(x), \neg\forall x(p(x) \rightarrow q(x))$
$\exists x p(x) \rightarrow \exists x q(x), \neg(p(a) \rightarrow q(a))$
$\exists x p(x) \rightarrow \exists x q(x), p(a), \neg q(a)$

$\overline{\text{L} \qquad \text{R}}$

$\neg\exists x p(x), p(a), \neg q(a)$
$\neg\exists x p(x), p(a), \neg q(a), \neg p(a)$
$\times$

R
$\exists x q(x), p(a), \neg q(a)$
$q(b), p(a), \neg q(a)$
$\odot$

Tableaux — A Well-Known Valid Formula

Prove the validity of a formula by showing that a tableau for its negation closes

$$\forall x(p(x) \rightarrow q(x)) \rightarrow (\exists x p(x) \rightarrow \exists x q(x))$$

Tableaux — A Well-Known Valid Formula

Prove the validity of a formula by showing that a tableau for its negation closes

$$\forall x(p(x) \to q(x)) \to (\exists x p(x) \to \exists x q(x))$$

$$\neg(\forall x(p(x) \to q(x)) \to (\exists x p(x) \to \exists x q(x)))$$
$$\forall x(p(x) \to q(x)), \neg(\exists x p(x) \to \exists x q(x))$$
$$\forall x(p(x) \to q(x)), \exists x p(x), \neg\exists x q(x)$$
$$\forall x(p(x) \to q(x)), p(a), \neg\exists x q(x)$$
$$\forall x(p(x) \to q(x)), p(a) \to q(a), p(a), \neg\exists x q(x), \neg q(a)$$

$\overline{\text{L} \qquad \text{R}}$

$$\forall x(p(x) \to q(x)), \neg p(a), p(a), \neg\exists x q(x), \neg q(a)$$
$\times$

R

$$\forall x(p(x) \to q(x)), q(a), p(a), \neg\exists x q(x), \neg q(a)$$
$\times$

Example Cont.

The available program considers only one γ-formula in each step...

```
?- tableau(~ ( all(X, p(X) => q(X)) =>
               (ex(X, p(X)) => ex(X, q(X))) )).

~(Ax1(p(x1) => q(x1)) => (Ex1p(x1) => Ex1q(x1)))
   Ax1(p(x1) => q(x1)),~(Ex1p(x1) => Ex1q(x1))
      Ex1p(x1),~Ex1q(x1),Ax1(p(x1) => q(x1))
         p(a1),~Ex1q(x1),Ax1(p(x1) => q(x1))
            ~q(a1),p(a1),Ax1(p(x1) => q(x1)),~Ex1q(x1)
               (p(a1) => q(a1)),~q(a1),p(a1),
                       ~Ex1q(x1),Ax1(p(x1) => q(x1))
                  ~p(a1),~q(a1),p(a1),~Ex1q(x1),
                          Ax1(p(x1) => q(x1)) Closed
                  q(a1),~q(a1),p(a1),~Ex1q(x1),
                          Ax1(p(x1) => q(x1)) Closed
Yes
```