Theory of Computer Science

B3. Propositional Logic III

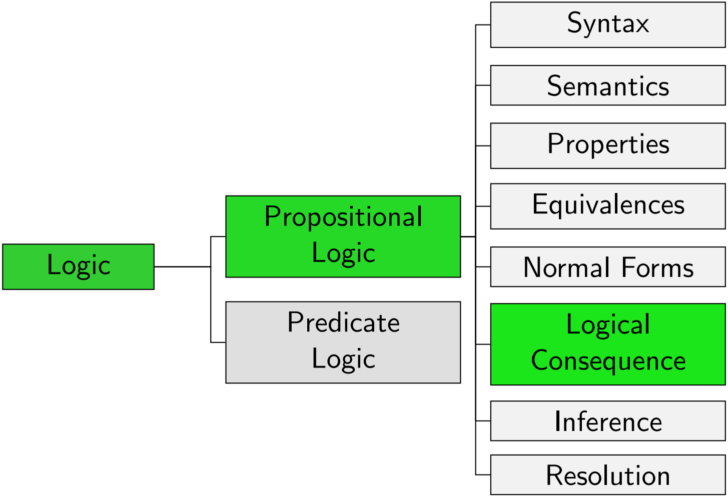
Gabriele Röger

University of Basel

February 26, 2020

Logical Consequences

Logic: Overview



Knowledge Bases: Example

If not DrinkBeer, then EatFish.
If EatFish and DrinkBeer,
then not EatIceCream.
If EatIceCream or not DrinkBeer,
then not EatFish.

$$
\begin{aligned}
KB = \{&(\neg \mathrm{DrinkBeer} \rightarrow \mathrm{EatFish}),\\
&((\mathrm{EatFish} \wedge \mathrm{DrinkBeer}) \rightarrow \neg \mathrm{EatIceCream}),\\
&((\mathrm{EatIceCream} \vee \neg \mathrm{DrinkBeer}) \rightarrow \neg \mathrm{EatFish})\}
\end{aligned}
$$

Exercise from U. Schöning: Logik für Informatiker
Picture courtesy of graur razvan ionut / FreeDigitalPhotos.net

Models for Sets of Formulas

Definition (Model for Knowledge Base)

Let KB be a knowledge base over A,
i. e., a set of propositional formulas over A.

A truth assignment $\mathcal{I}$ for A is a model for KB (written: $\mathcal{I} \models \text{KB}$)
if $\mathcal{I}$ is a model for every formula $\varphi \in \text{KB}$.

German: Wissensbasis, Modell

Properties of Sets of Formulas

A knowledge base KB is

- satisfiable if KB has at least one model
- unsatisfiable if KB is not satisfiable
- valid (or a tautology) if every interpretation is a model for KB
- falsifiable if KB is no tautology

German: erfüllbar, unerfüllbar, gültig, gültig/eine Tautologie, falsifizierbar

Example I

Which of the properties does $\mathrm{KB} = \{(A \land \neg B), \neg(B \lor A)\}$ have?

Example I

Which of the properties does $KB = \{(A \wedge \neg B), \neg(B \vee A)\}$ have?

KB is unsatisfiable:
For every model $\mathcal{I}$ with $\mathcal{I} \models (A \wedge \neg B)$ we have $\mathcal{I}(A) = 1$.
This means $\mathcal{I} \models (B \vee A)$ and thus $\mathcal{I} \not\models \neg(B \vee A)$.

Example I

Which of the properties does $\mathsf{KB} = \{(\mathsf{A} \wedge \neg \mathsf{B}), \neg(\mathsf{B} \vee \mathsf{A})\}$ have?

KB is unsatisfiable:
For every model $\mathcal{I}$ with $\mathcal{I} \models (\mathsf{A} \wedge \neg \mathsf{B})$ we have $\mathcal{I}(\mathsf{A}) = 1$.
This means $\mathcal{I} \models (\mathsf{B} \vee \mathsf{A})$ and thus $\mathcal{I} \not\models \neg(\mathsf{B} \vee \mathsf{A})$.

This directly implies that KB is falsifiable, not satisfiable and no tautology.

Example II

Which of the properties does

$$\begin{aligned}\mathsf{KB} = \{&(\neg\mathsf{DrinkBeer} \rightarrow \mathsf{EatFish}),\\ &((\mathsf{EatFish} \wedge \mathsf{DrinkBeer}) \rightarrow \neg\mathsf{EatIceCream}),\\ &((\mathsf{EatIceCream} \vee \neg\mathsf{DrinkBeer}) \rightarrow \neg\mathsf{EatFish})\}\text{ have?}\end{aligned}$$

Logical Consequences: Motivation

What's the secret of your long life?

I am on a strict diet: If I don't drink beer to a meal, then I always eat fish. Whenever I have fish and beer with the same meal, I abstain from ice cream. When I eat ice cream or don't drink beer, then I never touch fish.

Claim: the woman drinks beer to every meal.

How can we prove this?

Exercise from U. Schöning: Logik für Informatiker
Picture courtesy of graur razvan ionut/FreeDigitalPhotos.net

Logical Consequences

Definition (Logical Consequence)

Let KB be a set of formulas and φ a formula.

We say that KB logically implies φ (written as KB $\models \varphi$) if all models of KB are also models of φ.

also: KB logically entails φ, φ logically follows from KB, φ is a logical consequence of KB

German: KB impliziert φ logisch, φ folgt logisch aus KB, φ ist logische Konsequenz von KB

Logical Consequences

Definition (Logical Consequence)

Let KB be a set of formulas and φ a formula.

We say that KB logically implies φ (written as KB $\models \varphi$) if all models of KB are also models of φ.

also: KB logically entails φ, φ logically follows from KB, φ is a logical consequence of KB

German: KB impliziert φ logisch, φ folgt logisch aus KB, φ ist logische Konsequenz von KB

Attention: the symbol $\models$ is "overloaded": KB $\models \varphi$ vs. $\mathcal{I} \models \varphi$.

Logical Consequences

Definition (Logical Consequence)

Let KB be a set of formulas and φ a formula.

We say that KB logically implies φ (written as $\mathrm{KB} \models \varphi$)
if all models of KB are also models of φ.

also: KB logically entails φ, φ logically follows from KB,
φ is a logical consequence of KB

German: KB impliziert φ logisch, φ folgt logisch aus KB,
φ ist logische Konsequenz von KB

Attention: the symbol $\models$ is "overloaded": $\mathrm{KB} \models \varphi$ vs. $\mathcal{I} \models \varphi$.

What if KB is unsatisfiable or the empty set?

Logical Consequences: Example

Let $\varphi = \mathsf{DrinkBeer}$ and

$$\begin{aligned}\mathsf{KB} = \{&(\neg\mathsf{DrinkBeer} \rightarrow \mathsf{EatFish}),\\ &((\mathsf{EatFish} \wedge \mathsf{DrinkBeer}) \rightarrow \neg\mathsf{EatIceCream}),\\ &((\mathsf{EatIceCream} \vee \neg\mathsf{DrinkBeer}) \rightarrow \neg\mathsf{EatFish})\}.\end{aligned}$$

Show: $\mathsf{KB} \models \varphi$

Logical Consequences: Example

Let $\varphi = \text{DrinkBeer}$ and

$$\begin{aligned}\text{KB} = \{&(\neg\text{DrinkBeer} \rightarrow \text{EatFish}),\\ &((\text{EatFish} \wedge \text{DrinkBeer}) \rightarrow \neg\text{EatIceCream}),\\ &((\text{EatIceCream} \vee \neg\text{DrinkBeer}) \rightarrow \neg\text{EatFish})\}.\end{aligned}$$

Show: $\text{KB} \models \varphi$

Proof sketch.

Proof by contradiction: assume $\mathcal{I} \models \text{KB}$, but $\mathcal{I} \not\models \text{DrinkBeer}$.
Then it follows that $\mathcal{I} \models \neg\text{DrinkBeer}$.
Because $\mathcal{I}$ is a model of KB, we also have
$\mathcal{I} \models (\neg\text{DrinkBeer} \rightarrow \text{EatFish})$ and thus $\mathcal{I} \models \text{EatFish}$. (Why?)
With an analogous argumentation starting from
$\mathcal{I} \models ((\text{EatIceCream} \vee \neg\text{DrinkBeer}) \rightarrow \neg\text{EatFish})$
we get $\mathcal{I} \models \neg\text{EatFish}$ and thus $\mathcal{I} \not\models \text{EatFish}$. $\rightsquigarrow$ Contradiction!

Important Theorems about Logical Consequences

Theorem (Deduction Theorem)

$\mathsf{KB} \cup \{\varphi\} \models \psi$ *iff* $\mathsf{KB} \models (\varphi \rightarrow \psi)$

German: Deduktionssatz

Theorem (Contraposition Theorem)

$\mathsf{KB} \cup \{\varphi\} \models \neg\psi$ *iff* $\mathsf{KB} \cup \{\psi\} \models \neg\varphi$

German: Kontrapositionssatz

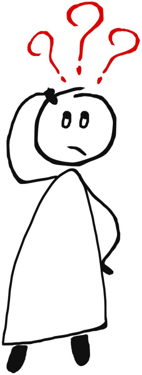
Theorem (Contradiction Theorem)

$\mathsf{KB} \cup \{\varphi\}$ *is unsatisfiable iff* $\mathsf{KB} \models \neg\varphi$

German: Widerlegungssatz

(without proof)

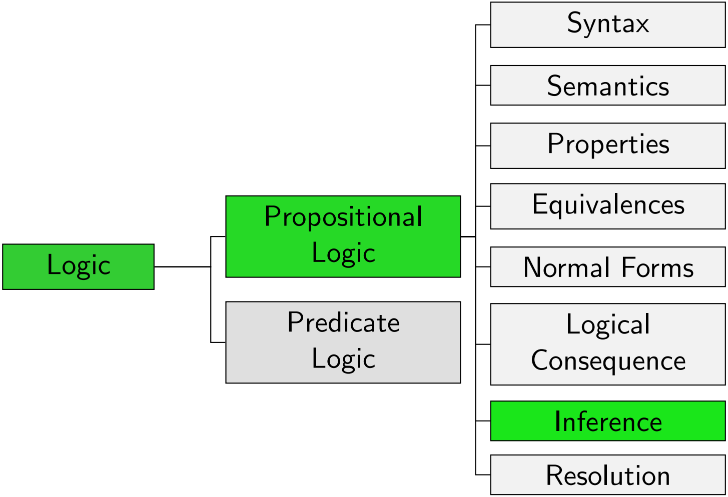
Questions



Questions?

Inference

Logic: Overview



Inference: Motivation

- up to now: proof of logical consequence with semantic arguments

Inference: Motivation

- up to now: proof of logical consequence with semantic arguments
- no general algorithm

Inference: Motivation

- up to now: proof of logical consequence with semantic arguments
- no general algorithm
- solution: produce with syntactic inference rules formulas that are logical consequences of given formulas.

Inference: Motivation

- up to now: proof of logical consequence with semantic arguments
- no general algorithm
- solution: produce with syntactic inference rules formulas that are logical consequences of given formulas.
- advantage: mechanical method can easily be implemented as an algorithm

Inference Rules

- Inference rules have the form

$$\frac{\varphi_1, \dots, \varphi_k}{\psi}.$$

German: Inferenzregel

Inference Rules

- Inference rules have the form

$$\frac{\varphi_1, \dots, \varphi_k}{\psi}.$$

- Meaning: "'Every model of $\varphi_1, \dots, \varphi_k$ is a model of ψ."'

German: Inferenzregel

Inference Rules

- Inference rules have the form

$$\frac{\varphi_1, \ldots, \varphi_k}{\psi}.$$

- Meaning: "'Every model of $\varphi_1, \ldots, \varphi_k$ is a model of ψ."'
- An axiom is an inference rule with $k = 0$.

German: Inferenzregel, Axiom

Inference Rules

- Inference rules have the form

$$\frac{\varphi_1, \dots, \varphi_k}{\psi}.$$

- Meaning: "'Every model of $\varphi_1, \dots, \varphi_k$ is a model of ψ."'
- An axiom is an inference rule with $k = 0$.
- A set of syntactic inference rules is called a calculus or proof system.

German: Inferenzregel, Axiom, Kalkül, Beweissystem

Some Inference Rules for Propositional Logic

Modus ponens $$\frac{\varphi,\ (\varphi \rightarrow \psi)}{\psi}$$

Some Inference Rules for Propositional Logic

Modus ponens $$\frac{\varphi,\ (\varphi \rightarrow \psi)}{\psi}$$

Modus tollens $$\frac{\neg\psi,\ (\varphi \rightarrow \psi)}{\neg\varphi}$$

Some Inference Rules for Propositional Logic

Modus ponens $$\frac{\varphi,\ (\varphi \rightarrow \psi)}{\psi}$$

Modus tollens $$\frac{\neg\psi,\ (\varphi \rightarrow \psi)}{\neg\varphi}$$

$\wedge$-elimination $$\frac{(\varphi \wedge \psi)}{\varphi} \qquad \frac{(\varphi \wedge \psi)}{\psi}$$

Some Inference Rules for Propositional Logic

Modus ponens $\dfrac{\varphi,\ (\varphi \rightarrow \psi)}{\psi}$

Modus tollens $\dfrac{\neg\psi,\ (\varphi \rightarrow \psi)}{\neg\varphi}$

$\wedge$-elimination $\dfrac{(\varphi \wedge \psi)}{\varphi}$ $\dfrac{(\varphi \wedge \psi)}{\psi}$

$\wedge$-introduction $\dfrac{\varphi,\ \psi}{(\varphi \wedge \psi)}$

Some Inference Rules for Propositional Logic

Modus ponens $$\frac{\varphi,\ (\varphi \to \psi)}{\psi}$$

Modus tollens $$\frac{\neg\psi,\ (\varphi \to \psi)}{\neg\varphi}$$

$\wedge$-elimination $$\frac{(\varphi \wedge \psi)}{\varphi} \qquad \frac{(\varphi \wedge \psi)}{\psi}$$

$\wedge$-introduction $$\frac{\varphi,\ \psi}{(\varphi \wedge \psi)}$$

$\vee$-introduction $$\frac{\varphi}{(\varphi \vee \psi)}$$

Some Inference Rules for Propositional Logic

Modus ponens $\dfrac{\varphi,\ (\varphi \rightarrow \psi)}{\psi}$

Modus tollens $\dfrac{\neg\psi,\ (\varphi \rightarrow \psi)}{\neg\varphi}$

$\wedge$-elimination $\dfrac{(\varphi \wedge \psi)}{\varphi}$ $\dfrac{(\varphi \wedge \psi)}{\psi}$

$\wedge$-introduction $\dfrac{\varphi,\ \psi}{(\varphi \wedge \psi)}$

$\vee$-introduction $\dfrac{\varphi}{(\varphi \vee \psi)}$

$\leftrightarrow$-elimination $\dfrac{(\varphi \leftrightarrow \psi)}{(\varphi \rightarrow \psi)}$ $\dfrac{(\varphi \leftrightarrow \psi)}{(\psi \rightarrow \varphi)}$

Derivation

Definition (Derivation)

A derivation or proof of a formula φ from a knowledge base KB is a sequence of formulas $\psi_1, \dots, \psi_k$ with

- $\psi_k = \varphi$ and
- for all $i \in \{1, \dots, k\}$:
  - $\psi_i \in \text{KB}$, or
  - ψ_i is the result of the application of an inference rule to elements from $\{\psi_1, \dots, \psi_{i-1}\}$.

German: Ableitung, Beweis

Derivation: Example

Example

Given: $\mathrm{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$
Task: Find derivation of $(S \land R)$ from KB.

Derivation: Example

Example

Given: $\text{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$
Task: Find derivation of $(S \land R)$ from KB.

1. P (KB)

Derivation: Example

Example

Given: $\mathrm{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$

Task: Find derivation of $(S \land R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)

Derivation: Example

Example

Given: $\text{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$

Task: Find derivation of $(S \land R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)

Derivation: Example

Example

Given: $KB = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$

Task: Find derivation of $(S \land R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)

Derivation: Example

Example

Given: $\mathrm{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \land R) \rightarrow S)\}$
Task: Find derivation of $(S \land R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)
5. R (1, 4, Modus ponens)

Derivation: Example

Example

Given: $KB = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \wedge R) \rightarrow S)\}$
Task: Find derivation of $(S \wedge R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)
5. R (1, 4, Modus ponens)
6. $(Q \wedge R)$ (3, 5, $\wedge$-introduction)

Derivation: Example

Example

Given: $\mathrm{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \wedge R) \rightarrow S)\}$
Task: Find derivation of $(S \wedge R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)
5. R (1, 4, Modus ponens)
6. $(Q \wedge R)$ (3, 5, $\wedge$-introduction)
7. $((Q \wedge R) \rightarrow S)$ (KB)

Derivation: Example

Example

Given: $\mathrm{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \wedge R) \rightarrow S)\}$
Task: Find derivation of $(S \wedge R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)
5. R (1, 4, Modus ponens)
6. $(Q \wedge R)$ (3, 5, $\wedge$-introduction)
7. $((Q \wedge R) \rightarrow S)$ (KB)
8. S (6, 7, Modus ponens)

Derivation: Example

Example

Given: $\text{KB} = \{P, (P \rightarrow Q), (P \rightarrow R), ((Q \wedge R) \rightarrow S)\}$
Task: Find derivation of $(S \wedge R)$ from KB.

1. P (KB)
2. $(P \rightarrow Q)$ (KB)
3. Q (1, 2, Modus ponens)
4. $(P \rightarrow R)$ (KB)
5. R (1, 4, Modus ponens)
6. $(Q \wedge R)$ (3, 5, $\wedge$-introduction)
7. $((Q \wedge R) \rightarrow S)$ (KB)
8. S (6, 7, Modus ponens)
9. $(S \wedge R)$ (8, 5, $\wedge$-introduction)

Correctness and Completeness

Definition (Correctness and Completeness of a Calculus)

We write $\mathrm{KB} \vdash_C \varphi$ if there is a derivation of φ from KB in calculus C.
(If calculus C is clear from context, also only $\mathrm{KB} \vdash \varphi$.)

A calculus C is correct if for all KB and φ
$\mathrm{KB} \vdash_C \varphi$ implies $\mathrm{KB} \models \varphi$.

A calculus C is complete if for all KB and φ
$\mathrm{KB} \models \varphi$ implies $\mathrm{KB} \vdash_C \varphi$.

Correctness and Completeness

Definition (Correctness and Completeness of a Calculus)

We write $\mathrm{KB} \vdash_C \varphi$ if there is a derivation of φ from KB in calculus C.
(If calculus C is clear from context, also only $\mathrm{KB} \vdash \varphi$.)

A calculus C is correct if for all KB and φ
$\mathrm{KB} \vdash_C \varphi$ implies $\mathrm{KB} \models \varphi$.

A calculus C is complete if for all KB and φ
$\mathrm{KB} \models \varphi$ implies $\mathrm{KB} \vdash_C \varphi$.

Consider calculus C, consisting of the derivation rules seen earlier.
Question: Is C correct?
Question: Is C complete?

German: korrekt, vollständig

Refutation-completeness

- We obviously want correct calculi.
- Do we always need a complete calculus?

Refutation-completeness

- We obviously want correct calculi.
- Do we always need a complete calculus?
- Contradiction theorem:
  $\mathrm{KB} \cup \{\varphi\}$ is unsatisfiable iff $\mathrm{KB} \models \neg\varphi$
- This implies that $\mathrm{KB} \models \varphi$ iff $\mathrm{KB} \cup \{\neg\varphi\}$ is unsatisfiable.
- We can reduce the general implication problem
  to a test of unsatisfiability.

Refutation-completeness

- We obviously want correct calculi.
- Do we always need a complete calculus?
- Contradiction theorem:
  $\mathrm{KB} \cup \{\varphi\}$ is unsatisfiable iff $\mathrm{KB} \models \neg\varphi$
- This implies that $\mathrm{KB} \models \varphi$ iff $\mathrm{KB} \cup \{\neg\varphi\}$ is unsatisfiable.
- We can reduce the general implication problem to a test of unsatisfiability.
- In calculi, we us the special symbol $\square$ for (provably) unsatisfiable formulas.

Refutation-completeness

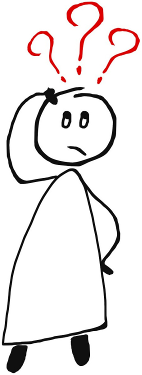
- We obviously want correct calculi.
- Do we always need a complete calculus?
- Contradiction theorem:
  $\mathrm{KB} \cup \{\varphi\}$ is unsatisfiable iff $\mathrm{KB} \models \neg\varphi$
- This implies that $\mathrm{KB} \models \varphi$ iff $\mathrm{KB} \cup \{\neg\varphi\}$ is unsatisfiable.
- We can reduce the general implication problem to a test of unsatisfiability.
- In calculi, we us the special symbol $\square$ for (provably) unsatisfiable formulas.

Definition (Refutation-Completeness)

A calculus C is refutation-complete if it holds for all unsatisfiable KB that $\mathrm{KB} \vdash_C \square$.

German: widerlegungsvollständig

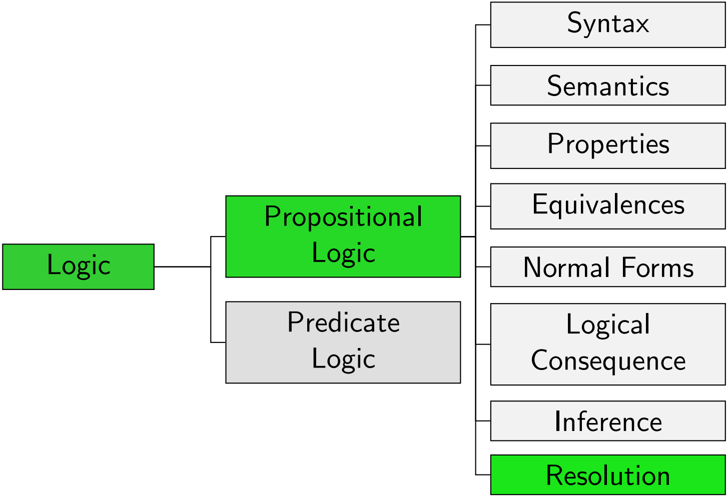
Questions



Questions?

Resolution Calculus

Logic: Overview



Resolution: Idea

- Resolution is a refutation-complete calculus for knowledge bases in conjunctive normal form.

Resolution: Idea

- Resolution is a refutation-complete calculus for knowledge bases in conjunctive normal form.
- Every knowledge base can be transformed into equivalent formulas in CNF.
  - Transformation can require exponential time.
  - Alternative: efficient transformation in equisatisfiable formulas (not part of this course)

Resolution: Idea

- Resolution is a refutation-complete calculus for knowledge bases in conjunctive normal form.
- Every knowledge base can be transformed into equivalent formulas in CNF.
  - Transformation can require exponential time.
  - Alternative: efficient transformation in equisatisfiable formulas (not part of this course)
- Show $\mathrm{KB} \models \varphi$ by derivability of $\mathrm{KB} \cup \{\neg\varphi\} \vdash_R \square$ with resolution calculus R.

Resolution: Idea

- Resolution is a refutation-complete calculus for knowledge bases in conjunctive normal form.
- Every knowledge base can be transformed into equivalent formulas in CNF.
  - Transformation can require exponential time.
  - Alternative: efficient transformation in equisatisfiable formulas (not part of this course)
- Show $\mathrm{KB} \models \varphi$ by derivability of $\mathrm{KB} \cup \{\neg\varphi\} \vdash_R \square$ with resolution calculus R.
- Resolution can require exponential time.
- This is probably the case for all refutation-complete proof methods. $\rightsquigarrow$ complexity theory

German: Resolution, erfüllbarkeitsäquivalent

Knowledge Base as Set of Clauses

Simplified notation of knowledge bases in CNF

- Formula in CNF as set of clauses
  (due to commutativity, idempotence, associativity of $\wedge$)
- Set of formulas as set of clauses
- Clause as set of literals
  (due to commutativity, idempotence, associativity of $\vee$)
- Knowledge base as set of sets of literals

Knowledge Base as Set of Clauses

Simplified notation of knowledge bases in CNF

- Formula in CNF as set of clauses
  (due to commutativity, idempotence, associativity of $\wedge$)
- Set of formulas as set of clauses
- Clause as set of literals
  (due to commutativity, idempotence, associativity of $\vee$)
- Knowledge base as set of sets of literals

Example

$$\mathrm{KB} = \{(P \vee P), ((\neg P \vee Q) \wedge (\neg P \vee R) \wedge (\neg P \vee Q) \wedge R), \\ ((\neg Q \vee \neg R \vee S) \wedge P)\}$$

as set of clauses:

Knowledge Base as Set of Clauses

Simplified notation of knowledge bases in CNF

- Formula in CNF as set of clauses
  (due to commutativity, idempotence, associativity of $\wedge$)
- Set of formulas as set of clauses
- Clause as set of literals
  (due to commutativity, idempotence, associativity of $\vee$)
- Knowledge base as set of sets of literals

Example

$$\mathrm{KB} = \{(P \vee P), ((\neg P \vee Q) \wedge (\neg P \vee R) \wedge (\neg P \vee Q) \wedge R),$$
$$((\neg Q \vee \neg R \vee S) \wedge P)\}$$

as set of clauses:

$$\Delta = \{\{P\}, \{\neg P, Q\}, \{\neg P, R\}, \{R\}, \{\neg Q, \neg R, S\}\}$$

Resolution Rule

The resolution calculus consists of a single rule, called resolution rule:

$$\frac{C_1 \cup \{L\},\ C_2 \cup \{\neg L\}}{C_1 \cup C_2},$$

where C_1 und C_2 are (possibly empty) clauses and L is an atomic proposition.

Resolution Rule

The resolution calculus consists of a single rule, called resolution rule:

$$\frac{C_1 \cup \{L\},\ C_2 \cup \{\neg L\}}{C_1 \cup C_2},$$

where C_1 und C_2 are (possibly empty) clauses and L is an atomic proposition.

If we derive the empty clause, we write $\square$ instead of $\{\}$.

Resolution Rule

The resolution calculus consists of a single rule, called resolution rule:

$$\frac{C_1 \cup \{L\},\ C_2 \cup \{\neg L\}}{C_1 \cup C_2},$$

where C_1 und C_2 are (possibly empty) clauses and L is an atomic proposition.

If we derive the empty clause, we write $\square$ instead of $\{\}$.

Terminology:

- L and $\neg L$ are the resolution literals,
- $C_1 \cup \{L\}$ and $C_2 \cup \{\neg L\}$ are the parent clauses, and
- $C_1 \cup C_2$ is the resolvent.

German: Resolutionskalkül, Resolutionsregel, Resolutionsliterale, Elternklauseln, Resolvent

Proof by Resolution

Definition (Proof by Resolution)

A proof by resolution of a clause D from a knowledge base Δ is a sequence of clauses $C_1, \ldots, C_n$ with

- $C_n = D$ and
- for all $i \in \{1, \ldots, n\}$:
  - $C_i \in \Delta$, or
  - C_i is resolvent of two clauses from $\{C_1, \ldots, C_{i-1}\}$.

If there is a proof of D by resolution from Δ, we say that D can be derived with resolution from Δ and write $\Delta \vdash_R D$.

Remark: Resolution is a correct, refutation-complete, but incomplete calculus.

German: Resolutionsbeweis, "mit Resolution aus Δ abgeleitet"

Proof by Resolution: Example

Proof by Resolution for Testing a Logical Consequence: Example

Given: $\mathrm{KB} = \{P, (P \rightarrow (Q \wedge R))\}$.
Show with resolution that $\mathrm{KB} \models (R \vee S)$.

Proof by Resolution: Example

Proof by Resolution for Testing a Logical Consequence: Example

Given: $\mathrm{KB} = \{P, (P \rightarrow (Q \land R))\}$.
Show with resolution that $\mathrm{KB} \models (R \lor S)$.

Three steps:

1. Reduce logical consequence to unsatisfiability.
2. Transform knowledge base into clause form (CNF).
3. Derive empty clause $\square$ with resolution.

Proof by Resolution: Example

Proof by Resolution for Testing a Logical Consequence: Example

Given: $\mathrm{KB} = \{P, (P \rightarrow (Q \wedge R))\}$.
Show with resolution that $\mathrm{KB} \models (R \vee S)$.

Three steps:

1. Reduce logical consequence to unsatisfiability.
2. Transform knowledge base into clause form (CNF).
3. Derive empty clause $\square$ with resolution.

Step 1: Reduce logical consequence to unsatisfiability.

Proof by Resolution: Example

Proof by Resolution for Testing a Logical Consequence: Example

Given: $\mathrm{KB} = \{P, (P \rightarrow (Q \wedge R))\}$.
Show with resolution that $\mathrm{KB} \models (R \vee S)$.

Three steps:

1. Reduce logical consequence to unsatisfiability.
2. Transform knowledge base into clause form (CNF).
3. Derive empty clause $\square$ with resolution.

Step 1: Reduce logical consequence to unsatisfiability.

$\mathrm{KB} \models (R \vee S)$ iff $\mathrm{KB} \cup \{\neg(R \vee S)\}$ is unsatisfiable.

Thus, consider
$\mathrm{KB}' = \mathrm{KB} \cup \{\neg(R \vee S)\} = \{P, (P \rightarrow (Q \wedge R)), \neg(R \vee S)\}$.

...

Proof by Resolution: Example (continued)

Proof by Resolution for Testing a Logical Consequence: Example

$\mathrm{KB}' = \{P, (P \rightarrow (Q \wedge R)), \neg(R \vee S)\}$.

Step 2: Transform knowledge base into clause form (CNF).

Proof by Resolution: Example (continued)

Proof by Resolution for Testing a Logical Consequence: Example

$\mathrm{KB}' = \{P, (P \rightarrow (Q \wedge R)), \neg(R \vee S)\}$.

Step 2: Transform knowledge base into clause form (CNF).

- P
  $\rightsquigarrow$ Clauses:$\{P\}$
- $P \rightarrow (Q \wedge R)) \equiv (\neg P \vee (Q \wedge R)) \equiv ((\neg P \vee Q) \wedge (\neg P \vee R))$
  $\rightsquigarrow$ Clauses:$\{\neg P, Q\}, \{\neg P, R\}$
- $\neg(R \vee S) \equiv (\neg R \wedge \neg S)$
  $\rightsquigarrow$ Clauses:$\{\neg R\}, \{\neg S\}$

Proof by Resolution: Example (continued)

Proof by Resolution for Testing a Logical Consequence: Example

$\mathrm{KB}' = \{P, (P \rightarrow (Q \wedge R)), \neg(R \vee S)\}$.

Step 2: Transform knowledge base into clause form (CNF).

- P
  $\rightsquigarrow$ Clauses: $\{P\}$
- $P \rightarrow (Q \wedge R)) \equiv (\neg P \vee (Q \wedge R)) \equiv ((\neg P \vee Q) \wedge (\neg P \vee R))$
  $\rightsquigarrow$ Clauses: $\{\neg P, Q\}, \{\neg P, R\}$
- $\neg(R \vee S) \equiv (\neg R \wedge \neg S)$
  $\rightsquigarrow$ Clauses: $\{\neg R\}, \{\neg S\}$

$\Delta = \{\{P\}, \{\neg P, Q\}, \{\neg P, R\}, \{\neg R\}, \{\neg S\}\}$

...

Proof by Resolution: Example (continued)

Proof by Resolution for Testing a Logical Consequence: Example

$\Delta = \{\{P\}, \{\neg P, Q\}, \{\neg P, R\}, \{\neg R\}, \{\neg S\}\}$

Step 3: Derive empty clause $\square$ with resolution.

- $C_1 = \{P\}$ (from Δ)
- $C_2 = \{\neg P, Q\}$ (from Δ)
- $C_3 = \{\neg P, R\}$ (from Δ)
- $C_4 = \{\neg R\}$ (from Δ)
- $C_5 = \{Q\}$ (from C_1 und C_2)
- $C_6 = \{\neg P\}$ (from C_3 und C_4)
- $C_7 = \square$ (from C_1 und C_6)

Note: There are shorter proofs. (For example?)

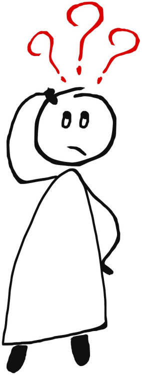
Another Example

Another Example for Resolution

Show with resolution, that KB $\models$ DrinkBeer, where

$$\begin{aligned} \mathrm{KB} = \{ & (\neg \mathrm{DrinkBeer} \rightarrow \mathrm{EatFish}), \\ & ((\mathrm{EatFish} \wedge \mathrm{DrinkBeer}) \rightarrow \neg \mathrm{EatIceCream}), \\ & ((\mathrm{EatIceCream} \vee \neg \mathrm{DrinkBeer}) \rightarrow \neg \mathrm{EatFish}) \}. \end{aligned}$$

Questions



Questions?

Summary

Summary

- knowledge base: set of formulas describing given information; satisfiable, valid etc. used like for individual formulas
- logical consequence $\mathrm{KB} \models \varphi$ means that φ is true whenever (= in all models where) KB is true
- A logical consequence $\mathrm{KB} \models \varphi$ allows to conclude that KB implies φ based on the semantics.
- A correct calculus supports such conclusions on the basis of purely syntactical derivations $\mathrm{KB} \vdash \varphi$.
- Complete calculi often not necessary: For many questions refutation-completeness is sufficient.
- The resolution calculus is correct and refutation-complete.

Further Topics

There are many aspects of propositional logic that we do not cover in this course.

- resolution strategies to make resolution as efficient as possible in practice,
- other proof systems, as for example tableaux proofs,
- algorithms for model construction, such as the Davis-Putnam-Logemann-Loveland (DPLL) algorithm.
  $\rightarrow$ Foundations of AI course