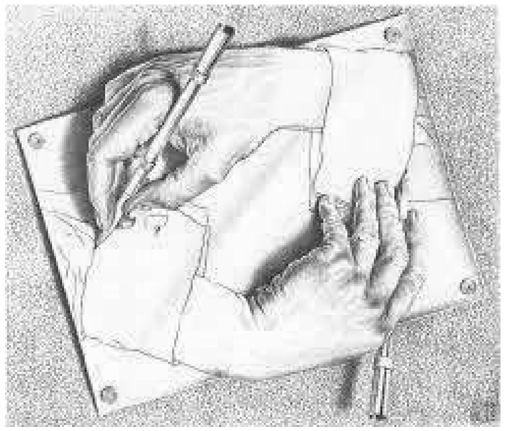
Logic

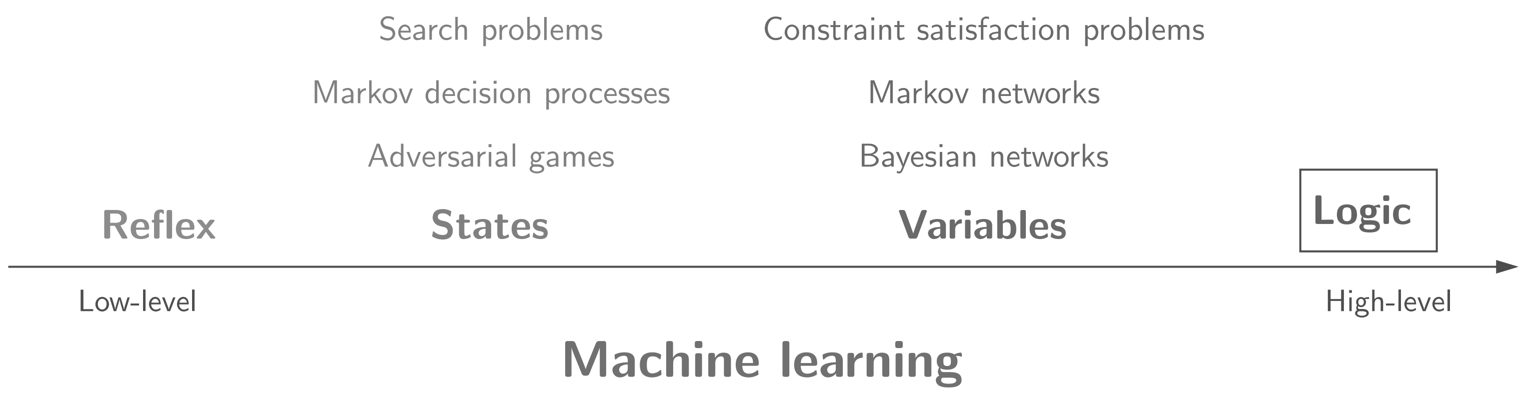


think and share

Question

If $X_1 + X_2 = 10$ and $X_1 - X_2 = 4$, what is X_1?

Course plan


Search problems
Markov decision processes
Adversarial games
Constraint satisfaction problems
Markov networks
Bayesian networks
Reflex
States
Variables
Logic
Low-level
High-level
Machine learning

Taking a step back

$$\begin{tikzcd} & & & & \text{question} \arrow[d] \\ \text{data} \arrow[r] & \boxed{\textbf{Learning}} \arrow[r] & \text{model} \arrow[r] & \boxed{\textbf{Inference}} \arrow[d] \\ & & & \text{answer} \end{tikzcd}$$

Examples: search problems, MDPs, games, CSPs, Bayesian networks

Modeling paradigms

State-based models: search problems, MDPs, games

Applications: route finding, game playing, etc.

Think in terms of **states, actions, and costs**

Variable-based models: CSPs, Bayesian networks

Applications: scheduling, tracking, medical diagnosis, etc.

Think in terms of **variables and factors**

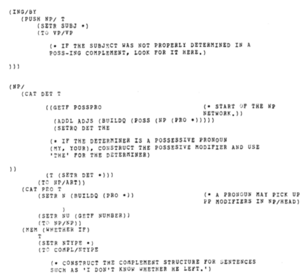
Logic-based models: propositional logic, first-order logic

Applications: theorem proving, verification, reasoning

Think in terms of **logical formulas and inference rules**

A historical note

- Logic was dominant paradigm in AI before 1990s



```
(ING/BY
   (PUSH NP/ T
       (SETR SUBJ •)
       (TO VP/VP

          (• IF THE SUBJECT WAS NOT PROPERLY DETERMINED IN A
          POSS-ING COMPLEMENT, LOOK FOR IT HERE.)

)))

(NP/
   (CAT DET T

         ((GETF POSSPRO                              (• START OF THE NP
                                                     NETWORK.))
           (ADDL ADJS (BUILDQ (POSS (NP (PRO •)))))
           (SETRQ DET THE

          (• IF THE DETERMINER IS A POSSESSIVE PRONOUN
          (MY, YOUR), CONSTRUCT THE POSSESIVE MODIFIER AND USE
          'THE' FOR THE DETERMINER)

))
         (T (SETR DET •)))
       (TO NP/ART))
   (CAT PRO T
       (SETR N (BUILDQ (PRO •))                      (• A PRONOUN MAY PICK UP
                                                     PP MODIFIERS IN NP/HEAD)
             )
       (SETR NU (GETF NUMBER))
       (TO NP/NP))
   (MEM (WHETHER IF)
       T
       (SETR NTYPE •)
       (TO COMPL/NTYPE

          (• CONSTRUCT THE COMPLEMENT STRUCTURE FOR SENTENCES
          SUCH AS 'I DON'T KNOW WHETHER HE LEFT.')
```

- Problem 1: deterministic, didn't handle **uncertainty** (probability addresses this)
- Problem 2: rule-based, didn't allow fine tuning from **data** (machine learning addresses this)
- Strength: provides **expressiveness** in a compact way

Motivation: smart personal assistant

Motivation: smart personal assistant

Tell information $\longrightarrow$

$\longleftarrow$ **Ask** questions

Use natural language!

[demo: `python nli.py`]

Need to:

- Digest **heterogenous** information
- Reason **deeply** with that information

Natural language

Example:

- A **dime** is better than a **nickel**.
- A **nickel** is better than a **penny**.
- Therefore, a **dime** is better than a **penny**.

Example:

- A **penny** is better than **nothing**.
- **Nothing** is better than **world peace**.
- Therefore, a **penny** is better than **world peace**???

Natural language is slippery...

Language

Language *is a mechanism for expression.*

Natural languages (informal):

English:	*Two divides even numbers.*
German:	*Zwei dividiert gerade Zahlen.*

Programming languages (formal):

Python:
```
def even(x):  return x % 2 == 0
```

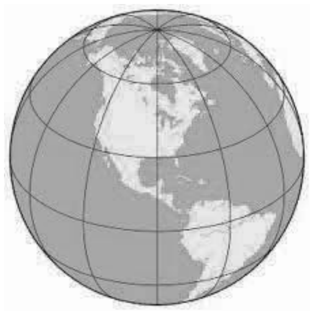
C++:
```
bool even(int x) { return x % 2 == 0; }
```

Logical languages (formal):

First-order-logic: $\forall x.\text{Even}(x) \rightarrow \text{Divides}(x, 2)$

Two goals of a logic language

- **Represent** knowledge about the world



- **Reason** with that knowledge

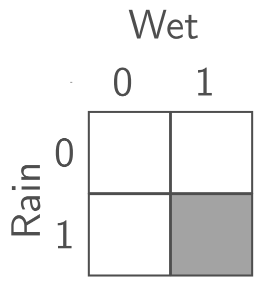
Ingredients of a logic

Syntax: defines a set of valid **formulas** (Formulas)

Example: Rain $\wedge$ Wet

Semantics: for each formula, specify a set of **models** (assignments / configurations of the world)

Example:



Inference rules: given f, what new formulas g can be added that are guaranteed to follow ($\frac{f}{g}$)?

Example: from Rain $\wedge$ Wet, derive Rain

Syntax versus semantics

Syntax: what are valid expressions in the language?

Semantics: what do these expressions mean?

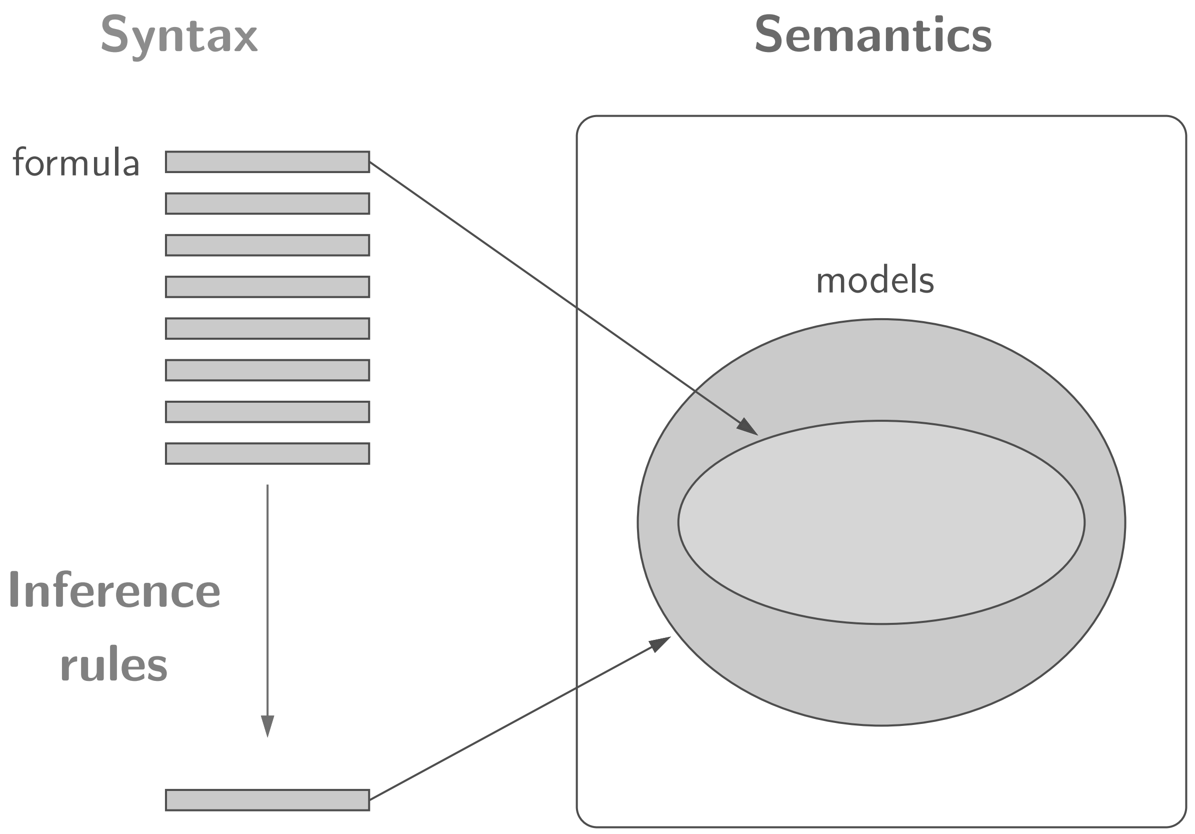
Different syntax, same semantics (5):

$$2 + 3 \quad \Leftrightarrow \quad 3 + 2$$

Same syntax, different semantics (1 versus 1.5):

$$3 \;/\; 2 \text{ (Python 2.7)} \quad \not\Leftrightarrow \quad 3 \;/\; 2 \text{ (Python 3)}$$

Propositional logic



Logics

- **Propositional logic with only Horn clauses**
- **Propositional logic**
- Modal logic
- **First-order logic with only Horn clauses**
- **First-order logic**
- Second-order logic
- ...

Key idea: tradeoff

Balance **expressivity** and **computational efficiency**.

Roadmap

Modeling

Propositional Logic Syntax

Propositional Logic Semantics

First-order Logic

Inference

Inference Rules

Propositional modus ponens

Propositional resolution

First-order modus ponens

First-order resolution

Lecture

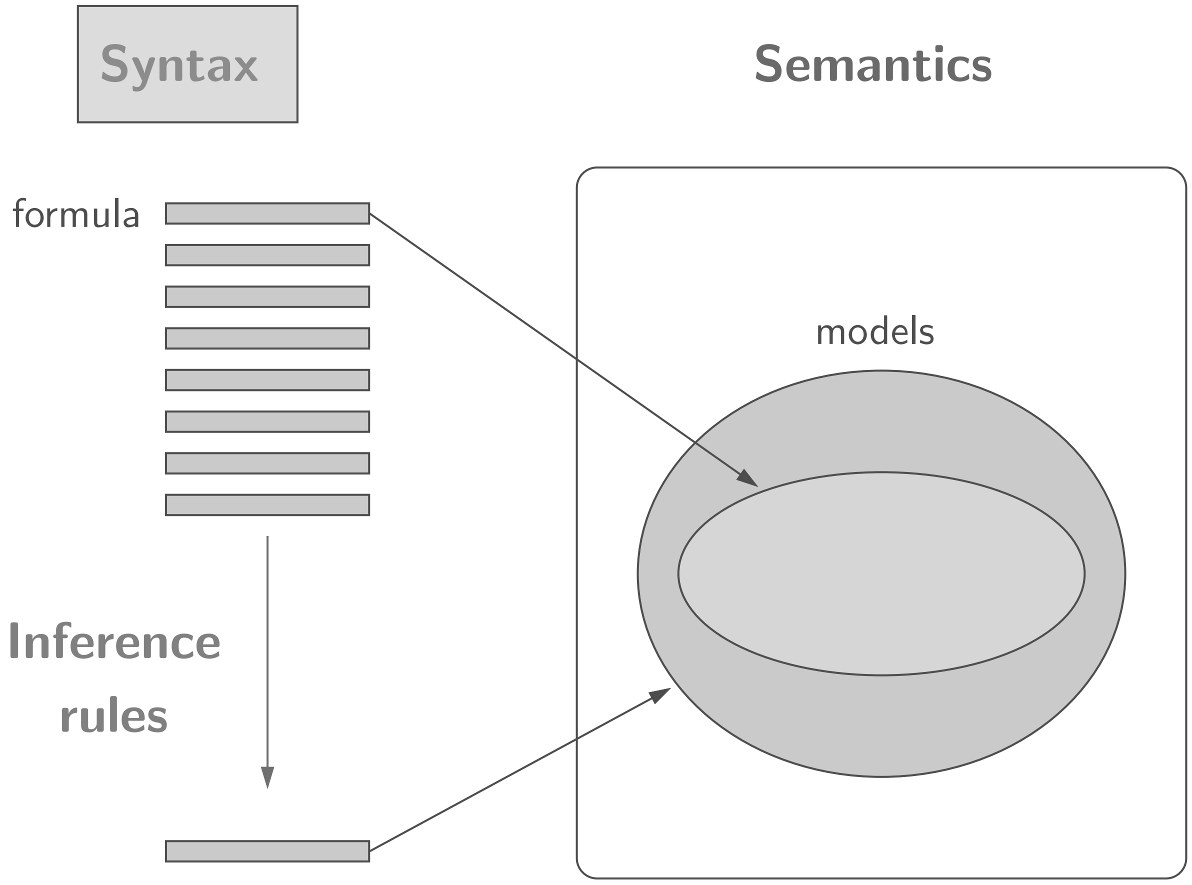
Overview

Propositional logic syntax

Propositional logic semantics

Inference rules

Propositional logic



Syntax of propositional logic

Propositional symbols (atomic formulas); A, B, C:

Logical connectives: $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

Build up formulas recursively—if f and g are formulas, so are the following:

- Negation: $\neg f$
- Conjunction: $f \wedge g$
- Disjunction: $f \vee g$
- Implication: $f \rightarrow g$
- Biconditional: $f \leftrightarrow g$

Syntax of propositional logic

- Formula: A
- Formula: $\neg A$
- Formula: $\neg B \to C$
- Formula: $\neg A \wedge (\neg B \to C) \vee (\neg B \vee D)$
- Formula: $\neg\neg A$
- Non-formula: $A \neg B$
- Non-formula: $A + B$

Syntax of propositional logic

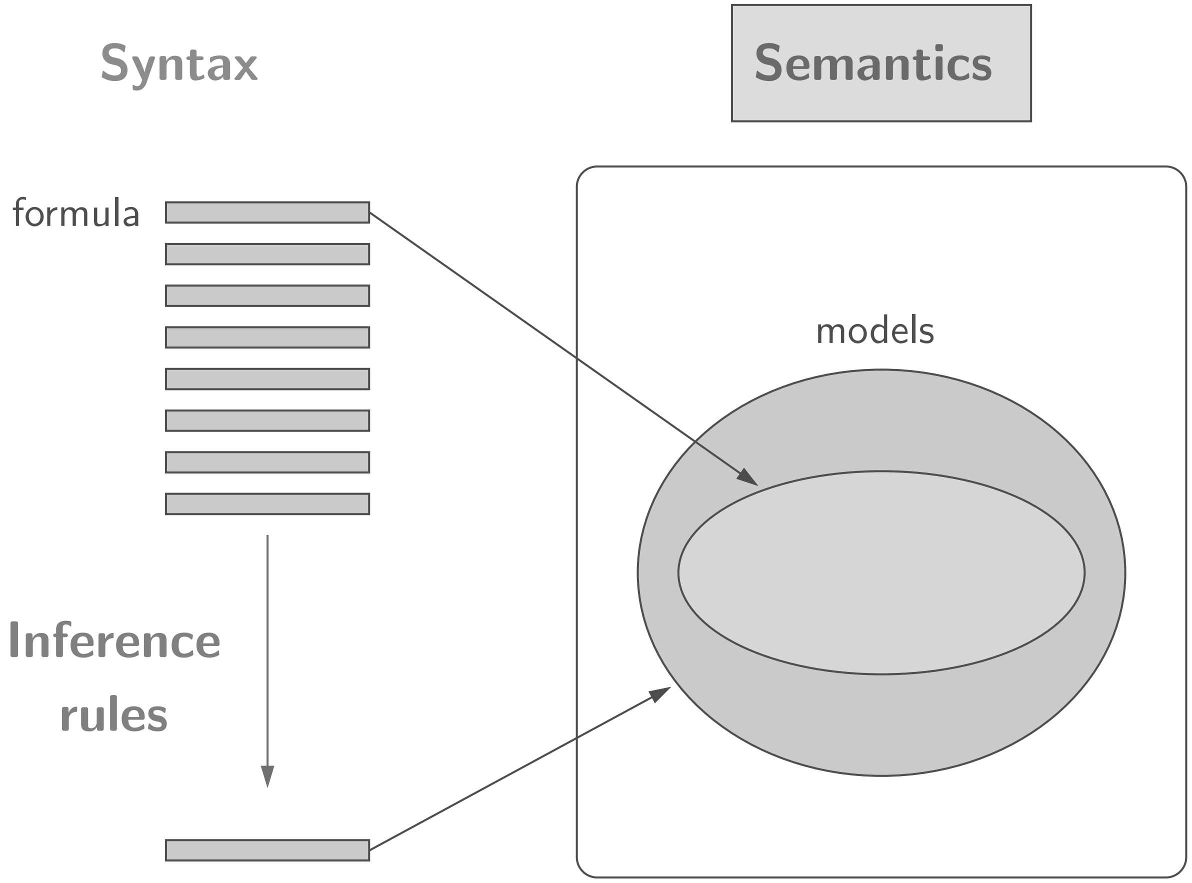
Key idea: syntax provides symbols

Formulas by themselves are just symbols (syntax).

No meaning yet (semantics)!

Lecture

Propositional logic



Model

Definition: model

A **model** w in propositional logic is an **assignment** of truth values to propositional symbols.

Example:

3 propositional symbols; A, B, C:

- $2^3 = 8$ possible models w:

$$\{A:0, B:0, C:0\}$$
$$\{A:0, B:0, C:1\}$$
$$\{A:0, B:1, C:0\}$$
$$\{A:0, B:1, C:1\}$$
$$\{A:1, B:0, C:0\}$$
$$\{A:1, B:0, C:1\}$$
$$\{A:1, B:1, C:0\}$$
$$\{A:1, B:1, C:1\}$$

Interpretation function

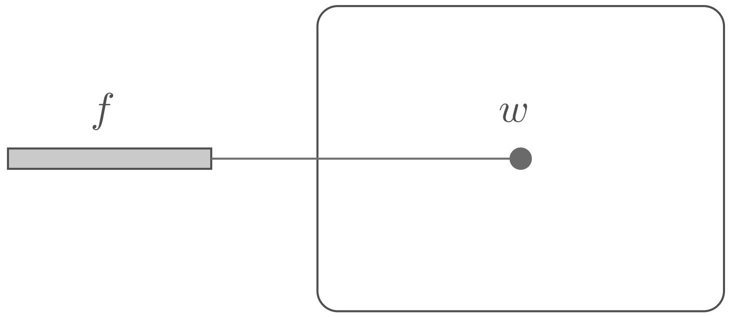
Definition: interpretation function

Let f be a formula.

Let w be a model.

An **interpretation function** $\mathcal{I}(f, w)$ returns:

- true (1) (say that w satisfies f)
- false (0) (say that w does not satisfy f)



Interpretation function: definition

Base case:

- For a propositional symbol p (e.g., A, B, C): $\mathcal{I}(p, w) = w(p)$

Recursive case:

- For any two formulas f and g, define:

$\mathcal{I}(f, w)$	$\mathcal{I}(g, w)$	$\mathcal{I}(\neg f, w)$	$\mathcal{I}(f \wedge g, w)$	$\mathcal{I}(f \vee g, w)$	$\mathcal{I}(f \rightarrow g, w)$	$\mathcal{I}(f \leftrightarrow g, w)$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Interpretation function: example

Example: interpretation function

Formula: $f = (\neg A \wedge B) \leftrightarrow C$

Model: $w = \{A : 1, B : 1, C : 0\}$

Interpretation:

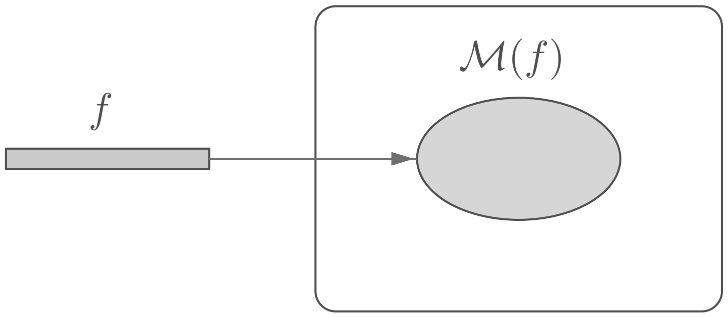
$$\begin{tikzcd}
 & & \boxed{\mathcal{I}((\neg A \wedge B) \leftrightarrow C, w) = 1} \arrow[dl, no head] \arrow[dr, no head] & \\
 & \boxed{\mathcal{I}(\neg A \wedge B, w) = 0} \arrow[dl, no head] \arrow[dr, no head] & & \boxed{\mathcal{I}(C, w) = 0} \\
\boxed{\mathcal{I}(\neg A, w) = 0} \arrow[d, no head] & & \boxed{\mathcal{I}(B, w) = 1} & \\
\boxed{\mathcal{I}(A, w) = 1} & & &
\end{tikzcd}$$

Formula represents a set of models

So far: each formula f and model w has an interpretation $\mathcal{I}(f, w) \in \{0, 1\}$

Definition: models

Let $\mathcal{M}(f)$ be the set of **models** w for which $\mathcal{I}(f, w) = 1$.



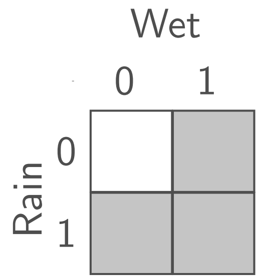
Models: example

Formula:

$$f = \text{Rain} \vee \text{Wet}$$

Models:

$$\mathcal{M}(f) =$$



Key idea: compact representation

A **formula** *compactly* represents a set of **models**.

Knowledge base

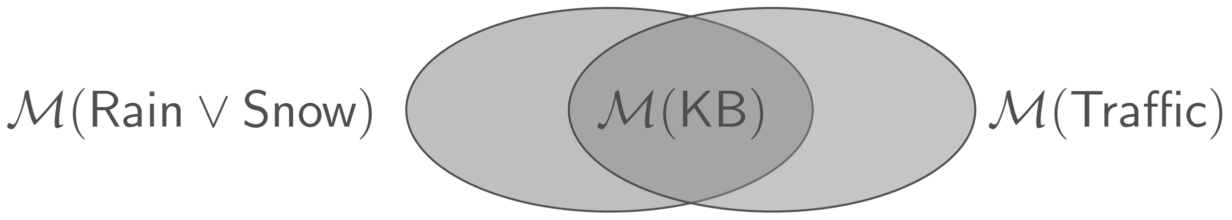
Definition: Knowledge base

A **knowledge base** KB is a set of formulas representing their conjunction / intersection:

$$\mathcal{M}(\text{KB}) = \bigcap_{f \in \text{KB}} \mathcal{M}(f).$$

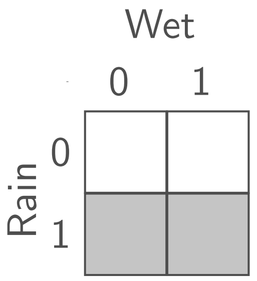
Intuition: KB specifies constraints on the world. $\mathcal{M}(\text{KB})$ is the set of all worlds satisfying those constraints.

Let $\text{KB} = \{\text{Rain} \vee \text{Snow}, \text{Traffic}\}$.

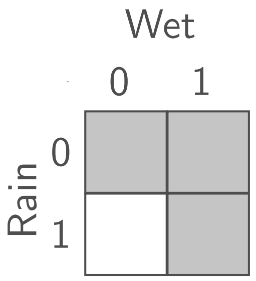

$\mathcal{M}(\text{Rain} \vee \text{Snow})$
$\mathcal{M}(\text{KB})$
$\mathcal{M}(\text{Traffic})$

Knowledge base: example

$\mathcal{M}(\text{Rain})$

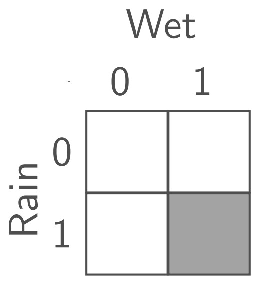


$\mathcal{M}(\text{Rain} \rightarrow \text{Wet})$



Intersection:

$\mathcal{M}(\{\text{Rain}, \text{Rain} \rightarrow \text{Wet}\})$



Adding to the knowledge base

Adding more formulas to the knowledge base:

$$\text{KB} \longrightarrow \text{KB} \cup \{f\}$$

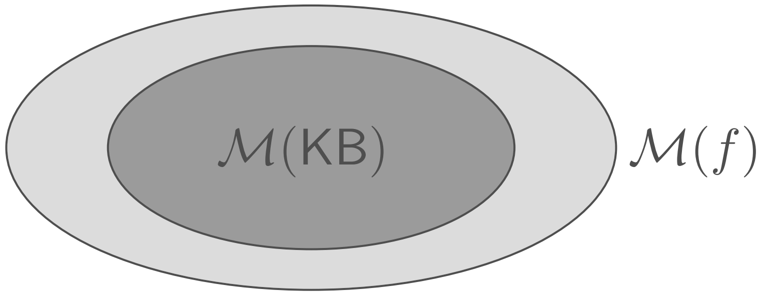
Shrinks the set of models:

$$\mathcal{M}(\text{KB}) \longrightarrow \mathcal{M}(\text{KB}) \cap \mathcal{M}(f)$$

How much does $\mathcal{M}(\textbf{KB})$ shrink?

[whiteboard]

Entailment



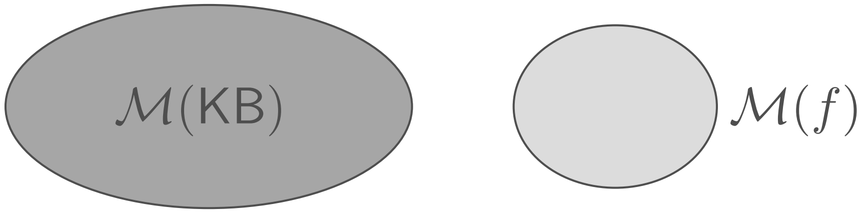
Intuition: f added no information/constraints (it was already known).

Definition: entailment

KB entails f (written KB $\models f$) iff $\mathcal{M}(\text{KB}) \subseteq \mathcal{M}(f)$.

Example: Rain $\land$ Snow $\models$ Snow

Contradiction



Intuition: f contradicts what we know (captured in KB).

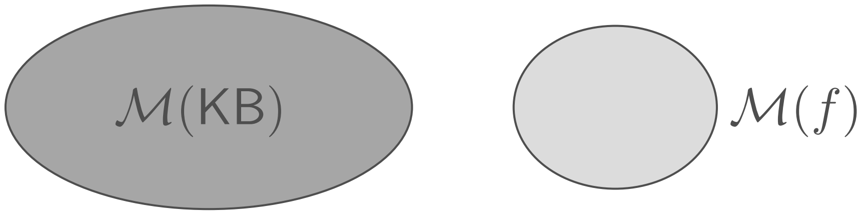
Definition: contradiction

KB contradicts f iff $\mathcal{M}(\text{KB}) \cap \mathcal{M}(f) = \emptyset$.

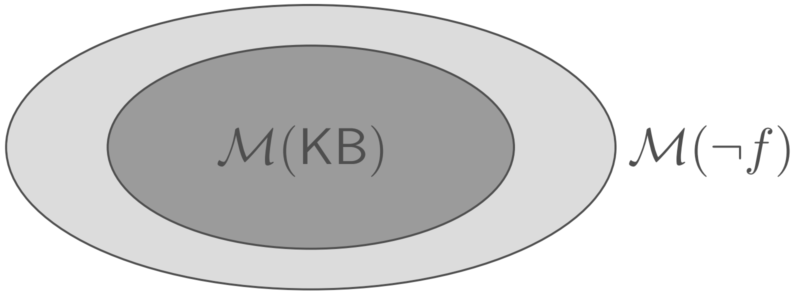
Example: Rain $\wedge$ Snow contradicts $\neg$Snow

Contradiction and entailment

Contradiction:



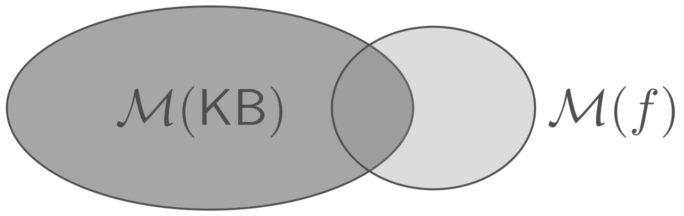
Entailment:



Proposition: contradiction and entailment

KB **contradicts** f iff KB **entails** $\neg f$.

Contingency



Intuition: f adds non-trivial information to KB

$$\emptyset \subsetneq \mathcal{M}(\text{KB}) \cap \mathcal{M}(f) \subsetneq \mathcal{M}(\text{KB})$$

Example: Rain and Snow

Tell operation

$$\text{Tell}[f] \longrightarrow \boxed{\text{KB}} \longrightarrow ?$$

Tell: *It is raining.*

$$\text{Tell}[\text{Rain}]$$

Possible responses:

- Already knew that: entailment (KB $\models f$)
- Don't believe that: contradiction (KB $\models \neg f$)
- Learned something new (update KB): contingent

Ask operation

$$\text{Ask}[f] \longrightarrow \boxed{\text{KB}} \longrightarrow ?$$

Ask: *Is it raining?*

$$\text{Ask}[\text{Rain}]$$

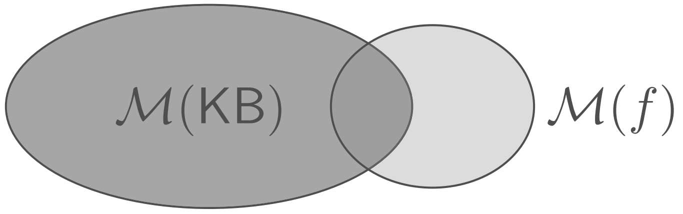
Possible responses:

- Yes: entailment ($\text{KB} \models f$)
- No: contradiction ($\text{KB} \models \neg f$)
- I don't know: contingent

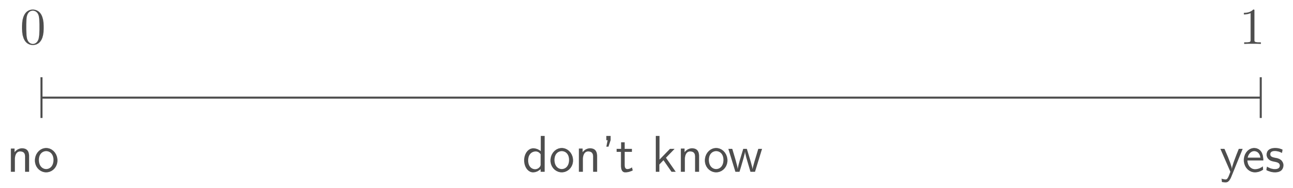
Digression: probabilistic generalization

Bayesian network: distribution over assignments (models)

w	$\mathbb{P}(W = w)$
{ A: 0, B: 0, C: 0 }	0.3
{ A: 0, B: 0, C: 1 }	0.1
...	...



$$\mathbb{P}(f \mid \mathsf{KB}) = \frac{\sum_{w \in \mathcal{M}(\mathsf{KB} \cup \{f\})} \mathbb{P}(W = w)}{\sum_{w \in \mathcal{M}(\mathsf{KB})} \mathbb{P}(W = w)}$$

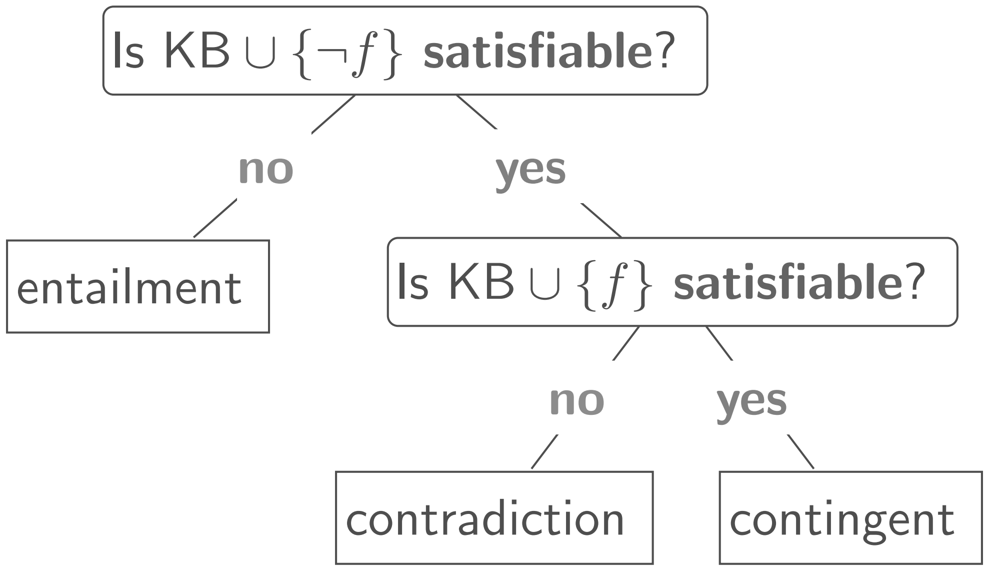


Satisfiability

Definition: satisfiability

A knowledge base KB is **satisfiable** if $\mathcal{M}(\text{KB}) \neq \emptyset$.

Reduce Ask$[f]$ and Tell$[f]$ to satisfiability:


Is KB $\cup \{\neg f\}$ **satisfiable**?
no
yes
entailment
Is KB $\cup \{f\}$ **satisfiable**?
no
yes
contradiction
contingent

Model checking

Checking satisfiability (SAT) in propositional logic is special case of solving CSPs!

Mapping:

propositional symbol	$\Rightarrow$	variable
formula	$\Rightarrow$	constraint
model	$\Leftarrow$	assignment

Model checking

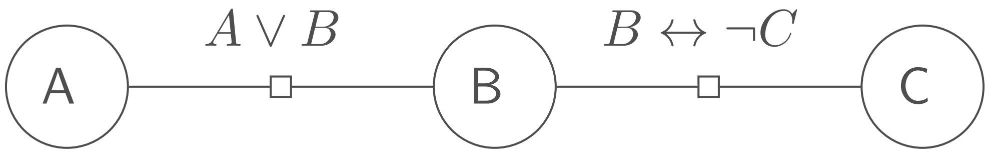
Example: model checking

KB $= \{A \vee B, B \leftrightarrow \neg C\}$

Propositional symbols (CSP variables):

$$\{A, B, C\}$$

CSP:



Consistent assignment (satisfying model):

$$\{A : 1, B : 0, C : 1\}$$

Model checking

Definition: model checking

Input: knowledge base KB

Output: exists satisfying model ($\mathcal{M}(\text{KB}) \neq \emptyset$)?

Popular algorithms:

- DPLL (backtracking search + pruning)
- WalkSat (randomized local search)

Next: Can we exploit the fact that factors are formulas?

Lecture

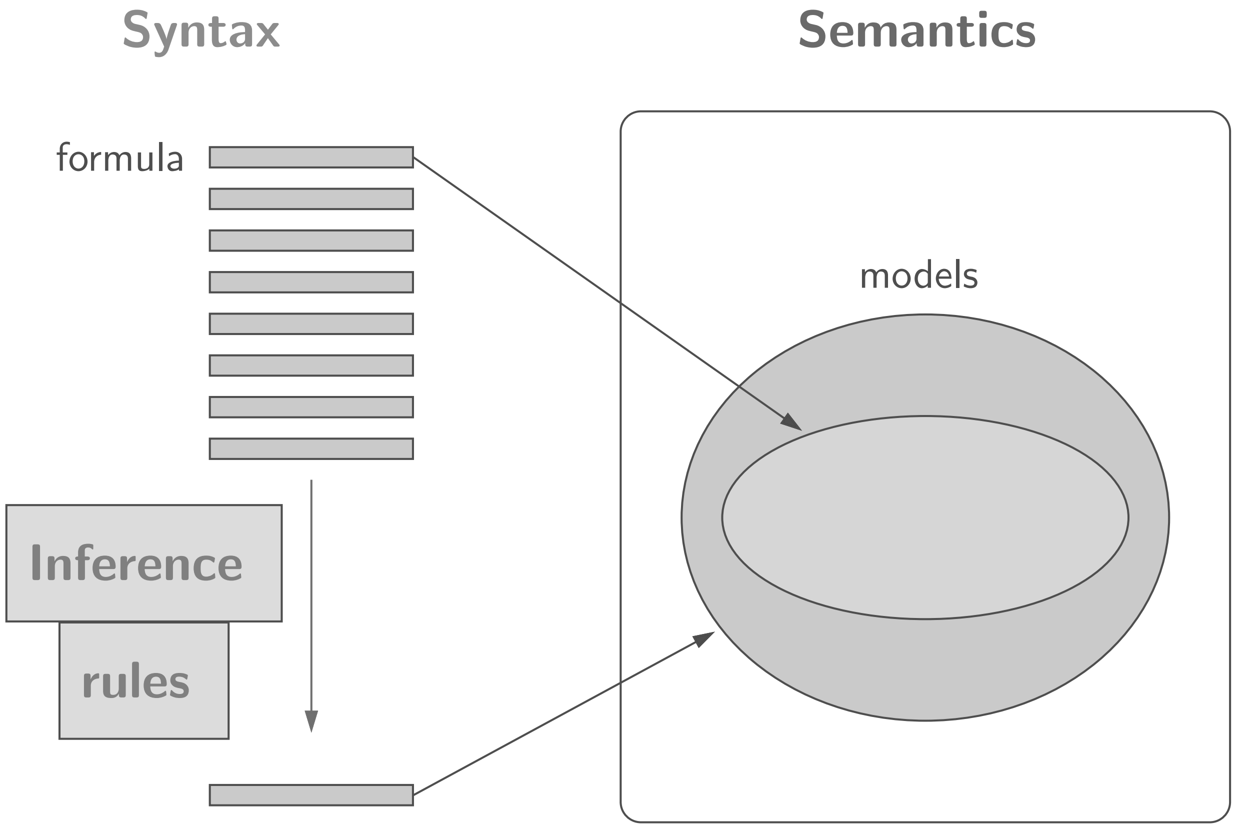
Overview

Propositional logic syntax

Propositional logic semantics

Inference rules

Propositional logic



Inference rules

Example of making an inference:

It is raining. (Rain)

If it is raining, then it is wet. (Rain $\rightarrow$ Wet)

Therefore, it is wet. (Wet)

$$\frac{\text{Rain}, \quad \text{Rain} \rightarrow \text{Wet}}{\text{Wet}} \qquad \frac{\text{(premises)}}{\text{(conclusion)}}$$

Definition: Modus ponens inference rule

For any propositional symbols p and q:

$$\frac{p, \quad p \rightarrow q}{q}$$

Inference framework

Definition: inference rule

If $f_1, \dots, f_k, g$ are formulas, then the following is an **inference rule**:

$$\frac{f_1, \quad \dots \quad , f_k}{g}$$

Key idea: inference rules

Rules operate directly on **syntax**, not on **semantics**.

Inference algorithm

Algorithm: forward inference

```
Input: set of inference rules Rules.
Repeat until no changes to KB:
    Choose set of formulas $f_1, \dots, f_k \in$ KB.
    If matching rule $\frac{f_1, \quad \dots \quad, f_k}{g}$ exists:
        Add $g$ to KB.
```

Definition: derivation

KB **derives/proves** f (KB $\vdash f$) iff f eventually gets added to KB.

Inference example

Example: Modus ponens inference

Starting point:

$$\text{KB} = \{\text{Rain}, \text{Rain} \rightarrow \text{Wet}, \text{Wet} \rightarrow \text{Slippery}\}$$

Apply modus ponens to Rain and Rain $\rightarrow$ Wet:

$$\text{KB} = \{\text{Rain}, \text{Rain} \rightarrow \text{Wet}, \text{Wet} \rightarrow \text{Slippery}, \text{Wet}\}$$

Apply modus ponens to Wet and Wet $\rightarrow$ Slippery:

$$\text{KB} = \{\text{Rain}, \text{Rain} \rightarrow \text{Wet}, \text{Wet} \rightarrow \text{Slippery}, \text{Wet}, \text{Slippery}\}$$

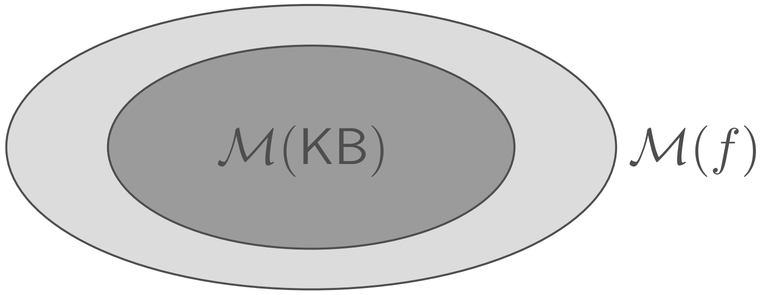
Converged.

Can't derive some formulas: $\neg\text{Wet}, \quad \text{Rain} \rightarrow \text{Slippery}$

Desiderata for inference rules

Semantics

Interpretation defines **entailed/true** formulas: KB $\models f$:



Syntax:

Inference rules **derive** formulas: KB $\vdash f$

How does $\{f : \text{KB} \models f\}$ relate to $\{f : \text{KB} \vdash f\}$?

Truth

$$\{f : \mathsf{KB} \models f\}$$

Soundness

Definition: soundness

A set of inference rules Rules is sound if:

$$\{f : \text{KB} \vdash f\} \subseteq \{f : \text{KB} \models f\}$$

Completeness

Definition: completeness

A set of inference rules Rules is complete if:

$$\{f : \mathsf{KB} \vdash f\} \supseteq \{f : \mathsf{KB} \models f\}$$

Soundness and completeness

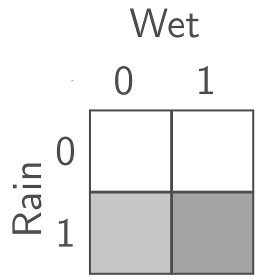
The truth, the whole truth, and nothing but the truth.

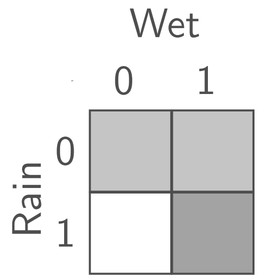
- **Soundness**: nothing but the truth
- **Completeness**: whole truth

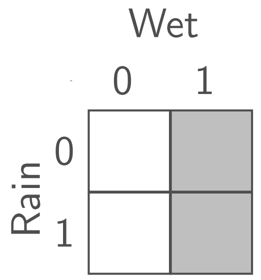
Soundness: example

Is $\frac{\text{Rain}, \quad \text{Rain} \to \text{Wet}}{\text{Wet}}$ (Modus ponens) sound?

$$\mathcal{M}(\text{Rain}) \quad \cap \quad \mathcal{M}(\text{Rain} \to \text{Wet}) \quad \subseteq ? \quad \mathcal{M}(\text{Wet})$$





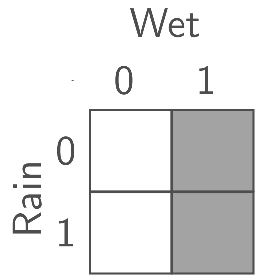


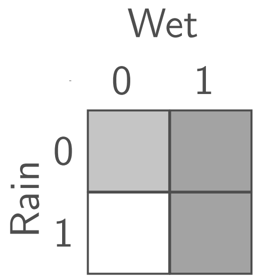
Sound!

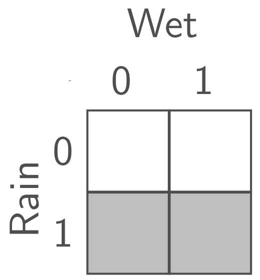
Soundness: example

Is $\dfrac{\text{Wet}, \quad \text{Rain} \rightarrow \text{Wet}}{\text{Rain}}$ sound?

$$\mathcal{M}(\text{Wet}) \quad \cap \quad \mathcal{M}(\text{Rain} \rightarrow \text{Wet}) \quad \subseteq? \quad \mathcal{M}(\text{Rain})$$







Unsound!

Completeness: example

Recall completeness: inference rules derive all entailed formulas (f such that KB $\models f$)

Example: Modus ponens is incomplete

Setup:

$$\text{KB} = \{\text{Rain}, \text{Rain} \vee \text{Snow} \rightarrow \text{Wet}\}$$

$$f = \text{Wet}$$

$$\text{Rules} = \left\{\frac{f, \quad f \rightarrow g}{g}\right\} \text{ (Modus ponens)}$$

Semantically: KB $\models f$ (f is entailed).

Syntactically: KB $\not\vdash f$ (can't derive f).

Incomplete!

Fixing completeness

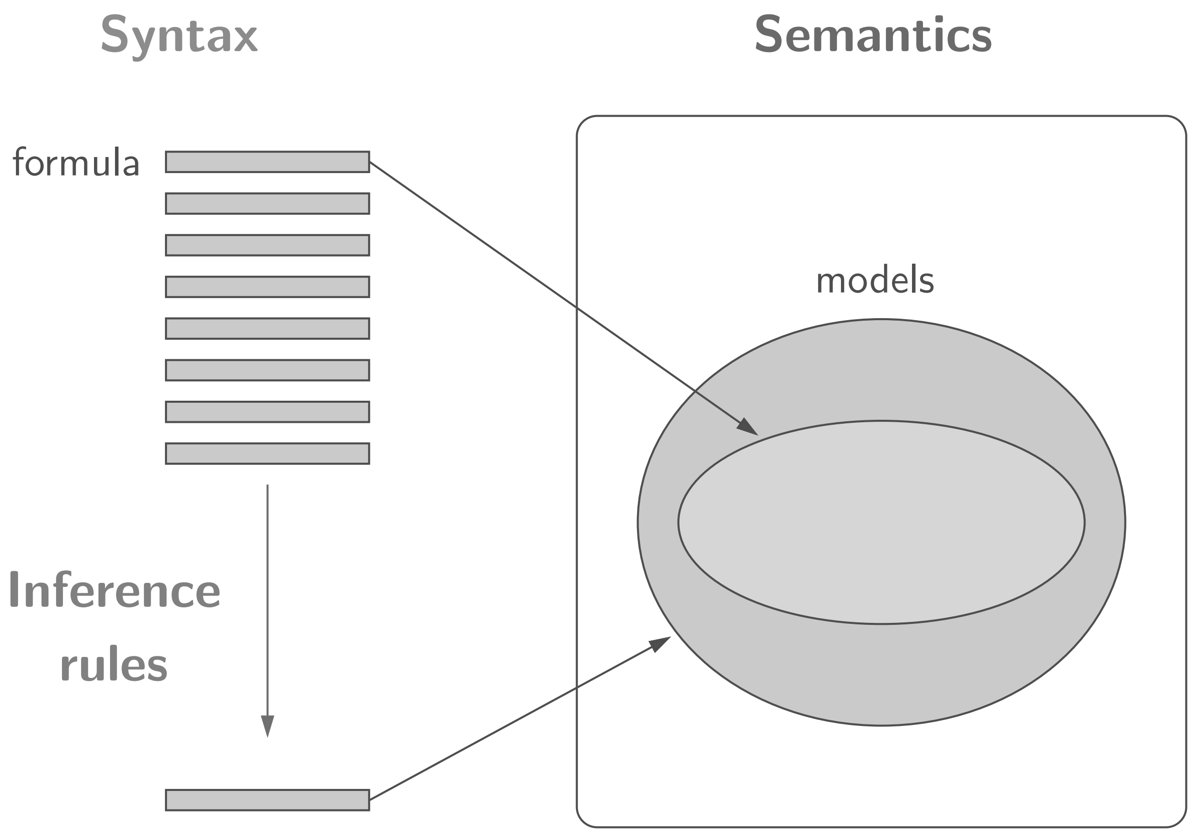
Option 1: Restrict the allowed set of formulas

$$\begin{tikzcd}\text{propositional logic} \arrow[d] \\ \text{propositional logic with only Horn clauses}\end{tikzcd}$$

Option 2: Use more powerful inference rules

$$\begin{tikzcd}\text{Modus ponens} \arrow[d] \\ \text{resolution}\end{tikzcd}$$

Summary


Syntax
Semantics
formula
models
Inference rules