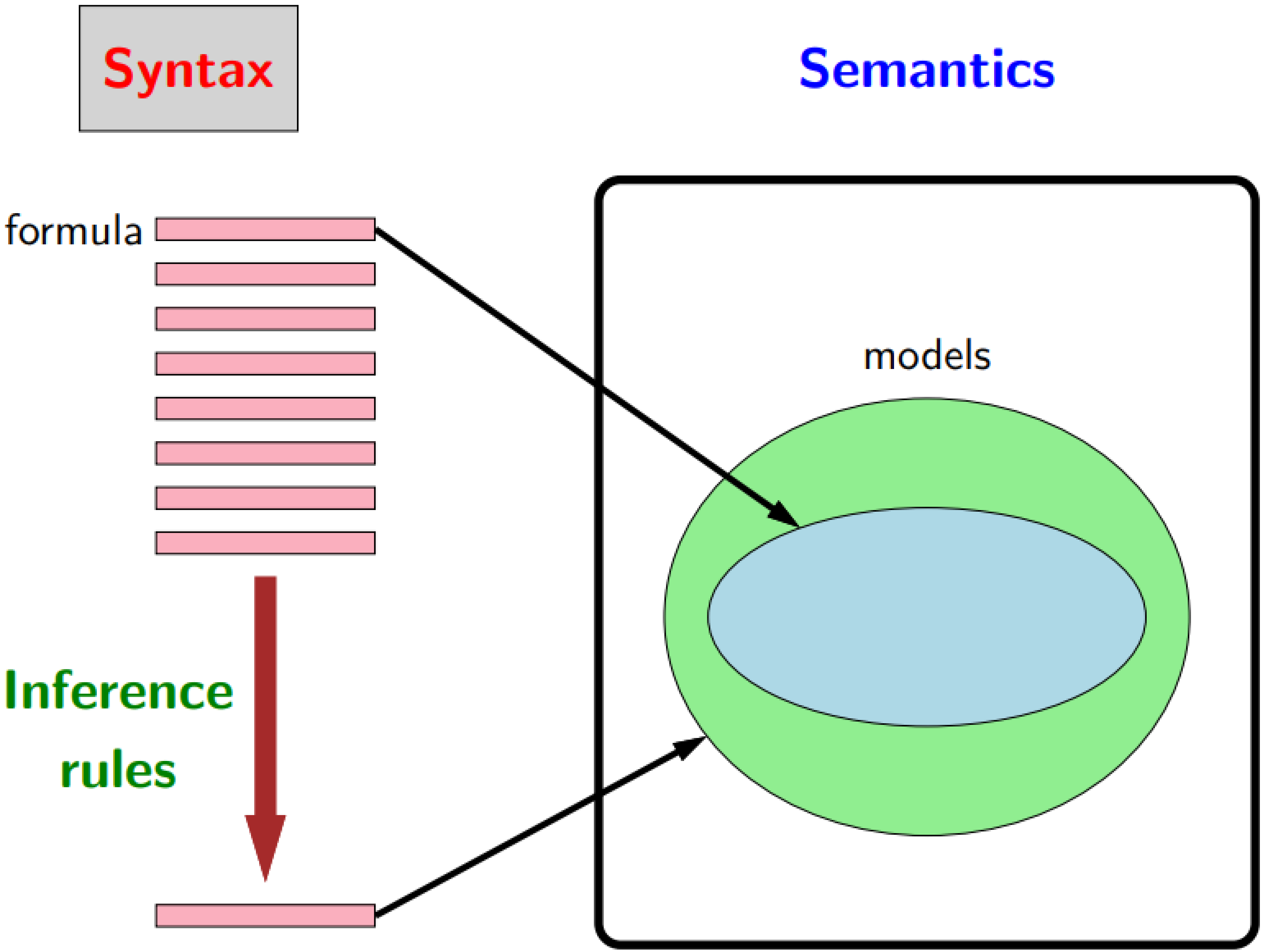
Logic

CS221 Section

By: Yi Wen

Logic

- Overview
  - Logic-based models
  - Motivation
  - Logical language
- Ingredients of a logic
  - Syntax: defines a set of valid formulas
  - Semantics: for each formula, specify a set of models
  - Inference rules: what new formulas can be added
- Propositional logic
- First-order logic


Syntax
Semantics
formula
models
Inference
rules

Propositional logic syntax

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: $f \leftrightarrow g$

Propositional logic semantics

Definition: model

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

Definition: interpretation function

Let f be a formula.

Let w be a model.

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

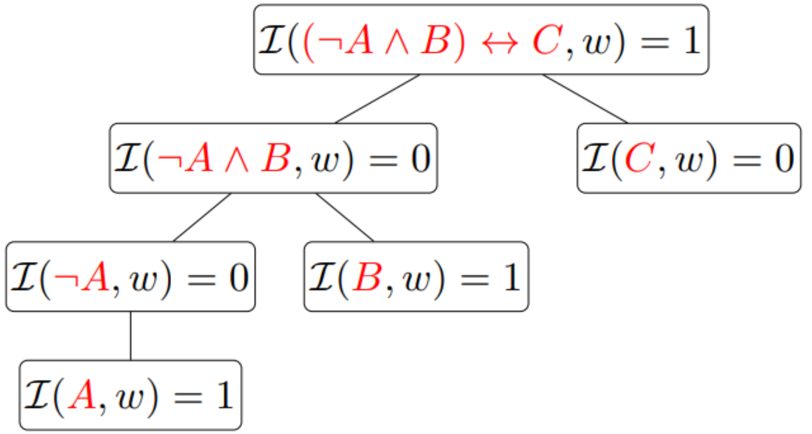
- true (1) (say that w satisfies f)
- false (0) (say that w does not satisfy f)

Example: interpretation function

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

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

Interpretation:



First-order logic syntax

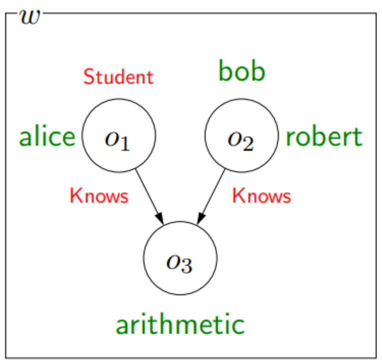
Terms (refer to objects):

- Constant symbol (e.g., arithmetic)
- Variable (e.g., x)
- Function of terms (e.g., $\text{Sum}(3, x)$)

Formulas (refer to truth values):

- Atomic formulas (atoms): predicate applied to terms (e.g., $\text{Knows}(x, \text{arithmetic})$)
- Connectives applied to formulas (e.g., $\text{Student}(x) \rightarrow \text{Knows}(x, \text{arithmetic})$)
- Quantifiers applied to formulas (e.g., $\forall x\, \text{Student}(x) \rightarrow \text{Knows}(x, \text{arithmetic})$)

First-order logic semantics



- Nodes are objects, labeled with constant symbols
- Directed edges are binary predicates, labeled with predicate symbols; unary predicates are additional node labels

Definition: model in first-order logic

A model w in first-order logic maps:

- constant symbols to objects

$w(\text{alice}) = o_1, w(\text{bob}) = o_2, w(\text{arithmetic}) = o_3$

- predicate symbols to tuples of objects

$w(\text{Knows}) = \{(o_1, o_3), (o_2, o_3), \dots\}$

First-order logic: Propositionalization

Knowledge base in first-order logic

$\text{Student}(\text{alice}) \wedge \text{Student}(\text{bob})$

$\forall x\, \text{Student}(x) \rightarrow \text{Person}(x)$

$\exists x\, \text{Student}(x) \wedge \text{Creative}(x)$

Knowledge base in propositional logic

$\text{Studentalice} \wedge \text{Studentbob}$

$(\text{Studentalice} \rightarrow \text{Personalice}) \wedge (\text{Studentbob} \rightarrow \text{Personbob})$

$(\text{Studentalice} \wedge \text{Creativealice}) \vee (\text{Studentbob} \wedge \text{Creativebob})$

Semantics: models and knowledge base

Definition: models

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

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.

Semantics: entailment, contradiction, contingency

Definition: entailment

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

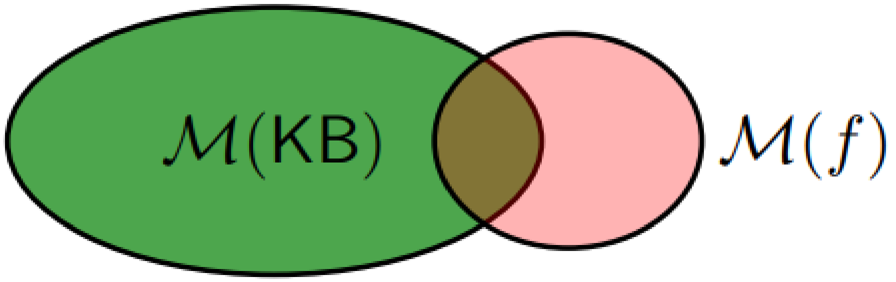
Definition: contradiction

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

Proposition: contradiction and entailment

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

Contingency



Intuition: f adds non-trivial information to KB

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

Semantics: Ask/Tell

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

Ask: *Is it raining?*

Ask[Rain]

Possible responses:

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

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

Tell: *It is raining.*

Tell[Rain]

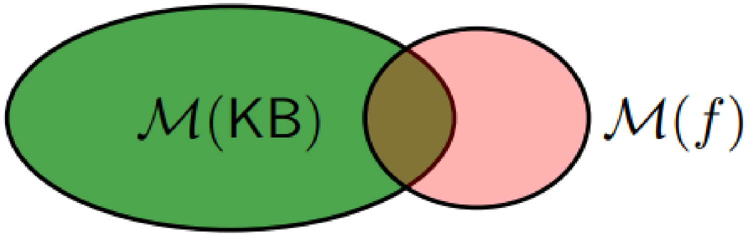
Possible responses:

- Already knew that
- Don't believe that
- Learned something new (update KB)

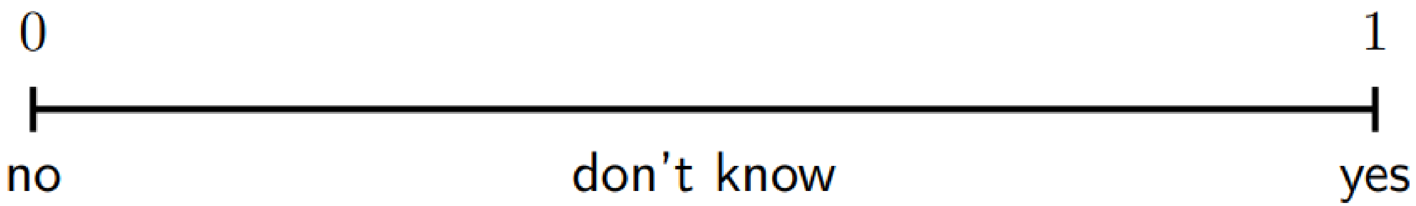
Semantics:
Digression - probabilistic generalization

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
...	...



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

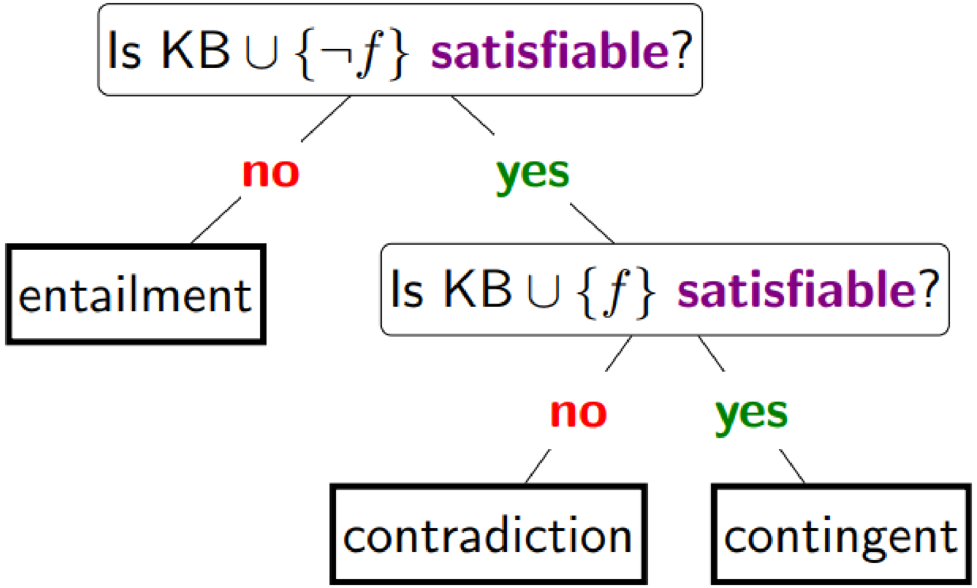


Semantics: satisfiability

Definition: satisfiability

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

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



Semantics: model checking

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

Definition: model checking

Input: knowledge base KB

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

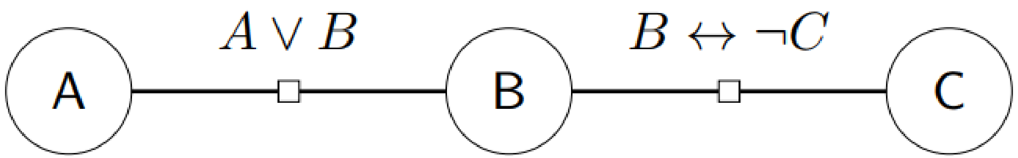
Example: model checking

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

Propositional symbols (CSP variables):

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

CSP:



Consistent assignment (satisfying model):

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

Inference rules

Definition: inference rule

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

$$\frac{f_1, \quad \ldots \quad , f_k}{g} \qquad \frac{\text{(premises)}}{\text{(conclusion)}}$$

Definition: Modus ponens inference rule

For any propositional symbols p and q:

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

Inference algorithm

Algorithm: forward inference

Input: set of inference rules Rules.

Repeat until no changes to KB:

Choose set of formulas $f_1, \ldots, f_k \in$ KB.

If matching rule $\frac{f_1, \quad \ldots \quad, f_k}{g}$ exists:

Add g to KB.

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.

Definition: derivation

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

Inference: soundness and completeness

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

Definition: soundness

A set of inference rules Rules is sound if:

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

Definition: completeness

A set of inference rules Rules is complete if:

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

Semantics

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

Syntax:

Inference rules **derive** formulas: $\text{KB} \vdash f$