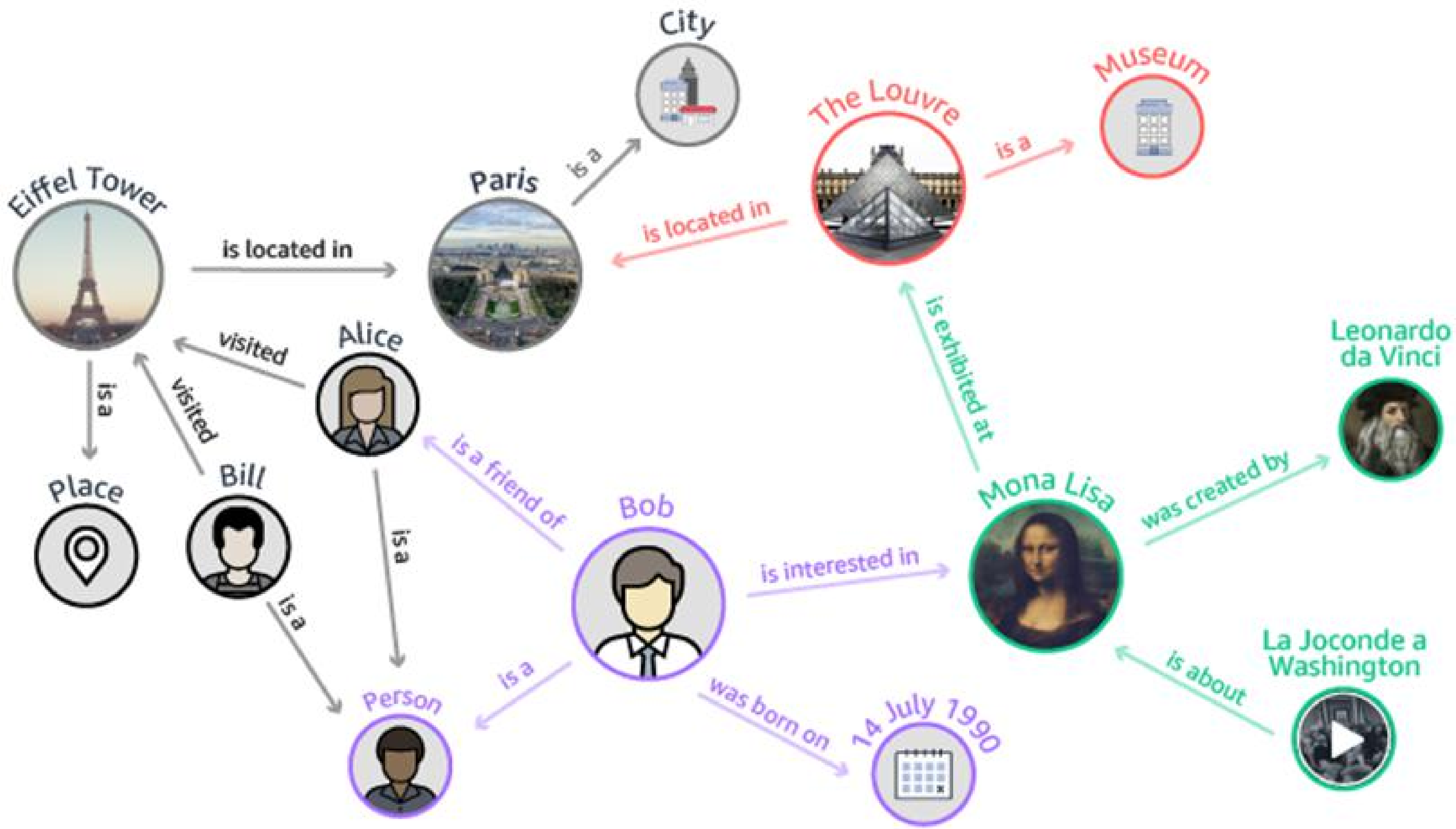
Artificial Intelligence

CS4365 --- Fall 2022

Knowledge Representation and Reasoning

Instructor: Yunhui Guo

Knowledge and Reasoning



Knowledge and Reasoning

- Knowledge:
  - the fact or condition of knowing something with familiarity gained through experience or association

- Reasoning:
  - the drawing of inferences or conclusions through the use of reason

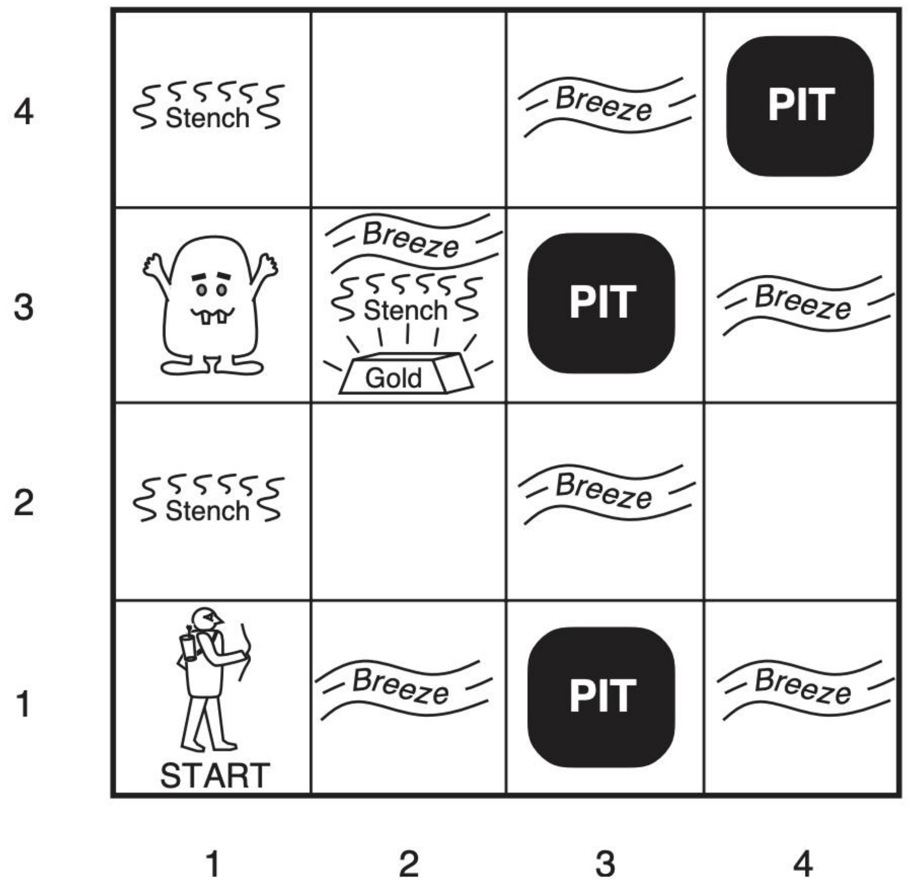
Knowledge + Reasoning → New Knowledge

The Wumpus World

- A decision-maker needs to represent knowledge of the world and reason with it in order to safely explore this world.

E.g. In the squares directly adjacent to a pit, the agent will perceive a Breeze

→ There a pit in [2, 2] or [3, 1] or both



Knowledge Representation

- Human intelligence relies on a lot of background knowledge (the more you know, the easier many tasks become / “knowledge is power”)

E.g. SEND + MORE = MONEY puzzle.

- Natural language understanding
  - Time flies like an arrow.
  - Fruit flies like bananas.

  - The spirit is willing but the flesh is weak. (English)
  - The vodka is good but the meat is rotten. (Russian)
- Or: Plan a trip to L.A.

Knowledge Representation

Q. How did we encode (domain) knowledge so far?

For search problems?

Fine for limited amounts of knowledge / well-defined domains.

Otherwise: knowledge-based systems approach

Knowledge-Based Systems / Agents

Key components

- knowledge base: a set of sentences expressed in some knowledge representation language
- Inference / reasoning mechanisms to query what is known and to derive new information or make decisions

Knowledge-Based Systems / Agents

- Natural candidate: logical language (propositional / first-order) combined with a logical inference mechanism

- How close to human thought?

- In any case, appears reasonable strategy for machines

Logic

- Logic:
  - defines a formal language for logical reasoning
- It gives us a tool that helps us to understand how to construct a valid argument
- Logic defines:
  - the meaning of statements
  - the rules of logical inference

Logic as a Knowledge Representation

Three components:

- syntax: specifies which sentences can be constructed in a given formal logic
    - E.g. $x + y = 4$
- semantics: specifies what a sentence means
    - $x + y = 4$ is True if $x = 2$ and $y = 2$
- proof theory: a set of general purpose rules that allow efficient derivation of new information from the sentences in the knowledge base

Logic as a Knowledge Representation

Model: a truth assignment to every propositonal symbol

Logic entailment:

A sentence follows logically from another sentence:

$$\alpha \models \beta$$

In every model in which α is true, β is also true.

Logic as a Knowledge Representation

- Logic inference:
  - Given a knowledge base KB and a sentence α
  - Does a KB semantically entail α? $KB \models \alpha$

One possible approach:

Model Checking: enumerate all the possible models to check if α is true in all models in which KB is true

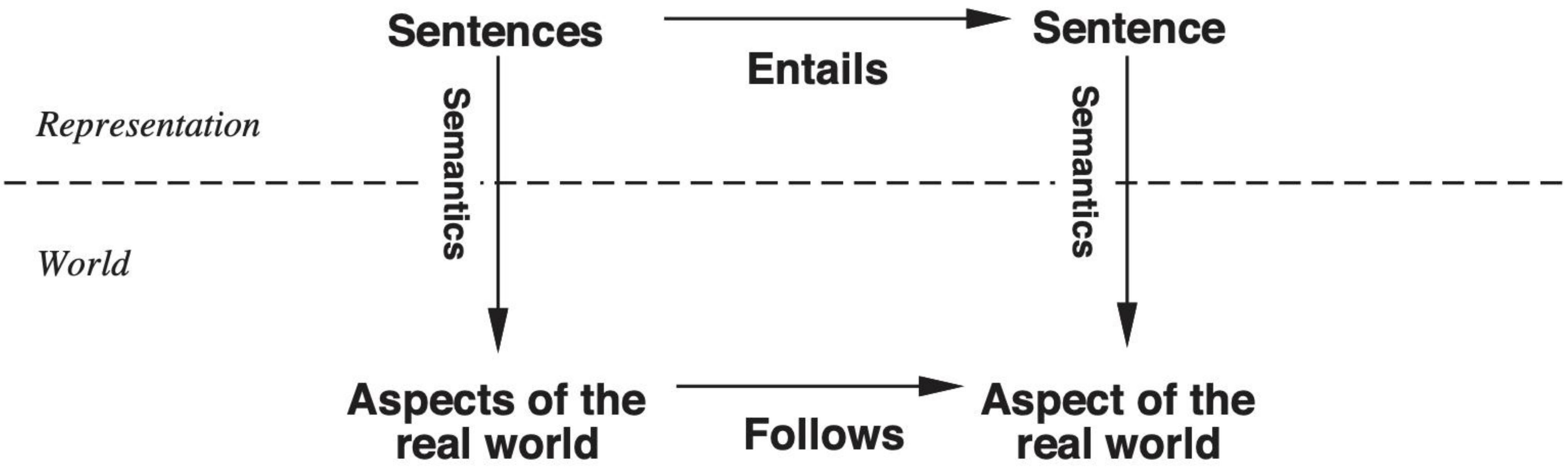
Logic as a Knowledge Representation

Proof theory:

Sound: An inference algorithm that derives only entailed sentences

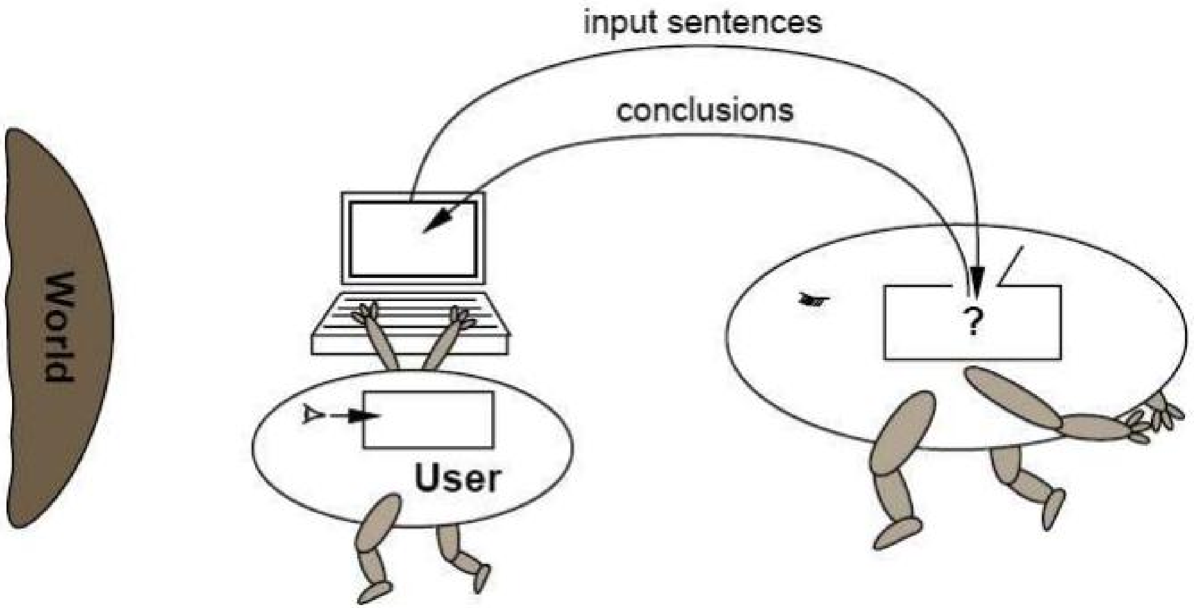
Complete: an inference algorithm is complete if it can derive any sentence that is entailed

Connecting Sentences to the Real World



- Logical reasoning should ensure that the new configurations represent aspects of the world that actually follow from the aspects that the old configurations represent.

Tenuous Link to Real World



- All computer has are sentences (hopefully about the world).

KR Language: Propositional Logic

- Go back to 3rd century B.C. studied by Stoic school of philosophy

- Real development began in the mid-19th century and was initiated by the English mathematician G. Boole

- The classical propositional calculus was first formulated as a formal axiomatic system by the eminent German logician G. Frege in 1879.

KR Language: Propositional Logic

- The simplest logic
- Definition
  - A proposition is a statement that is either true or false.

- Example:
  - 5 + 2 = 8 (F)
  - It is raining today
    - (either T or F)

KR Language: Propositional Logic

- Literal: an atomic formula or its negation
  - Positive literal: P, Q
  - Negative literal: ¬P, ¬Q

- Syntax: build sentences from atomic propositions, using connectives:
  - $\wedge$: and
  - $\vee$: or
  - ¬ : not
  - ⇒ : implies
  - ⇔ : equivalence (biconditional)

KR Language: Propositional Logic

Syntax: build sentences from atomic propositions, using connectives $\wedge$, $\vee$, $\neg$, $\Rightarrow$, $\Leftrightarrow$

(and / or / not / implies / equivalence (biconditional))

E.g.: $\neg P$

$Q \wedge R$

$(\neg P \vee (Q \wedge R)) \Rightarrow S$

KR Language: Propositional Logic

- Clause: a disjunction of literals

  E.g.: $Q \lor R$

- Conjunctive normal form (CNF): a conjunction of clauses

  E.g.: $(Q \lor R) \land (P \lor R)$

- Every formula can be equivalently written as a formula in conjunctive normal form

  $(Q \land R) \lor P \quad \rightarrow \quad (Q \lor P) \land (R \lor P)$

Semantics

Semantics specifies what something means.

In propositional logic, the semantics (i.e., meaning) of a sentence is the set of interpretations (i.e., truth assignments) in which the sentence evaluates to True.

Example:

The semantics of the sentence $P \vee Q \Rightarrow R$ is

- P is True, Q is True, R is True
- P is True , Q is False, R is True
- P is False , Q is True , R is True
- P is False , Q is False , R is True
- P is False , Q is False , R is False

Interpretations: The Key to Semantics

An interpretation is a logician's word for "truth assignment"

- Given 3 propositional symbols P, Q, R, there are 8 interpretations.
- Given n propositional symbols P_1, P_2,... P_n, there are 2^n interpretations

In propositional logic:

- an interpretation is a mapping from propositional symbols to truth values.
- the meaning of a sentence is the set of interpretations in which the sentence evaluates to True

How to evaluate a sentence under a given interpretation?

Evaluating a sentence under interpretation I

We can evaluate a sentence using a truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Evaluating a sentence under interpretation I

We can evaluate a sentence using a truth table

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	*false*	*true*	*false*	*false*	*true*	*true*
false	*true*	*true*	*false*	*true*	*true*	*false*
true	*false*	*false*	*false*	*true*	*false*	*false*
true	*true*	*false*	*true*	*true*	*true*	*true*

Note: $\Longrightarrow$ is somewhat counterintuitive

What's the true value of "5 is even implies Sam is smart"

If P is True, then I claim Q is True

Three Important Concepts

- Logic Equivalence
- Validity
- Satisfiability

Logic Equivalence

- Two sentences are equivalent if they are true in the same set of models.

- We write this as $\alpha \equiv \beta$. $\alpha \equiv \beta$ if and only if $\alpha \vDash \beta$ and $\beta \vDash \alpha$

For example:

I. If Lisa is in Denmark, then she is in Europe

II. If Lisa is not in Europe, then she is not in Denmark

Logic Equivalence

- $(\alpha \land \beta) \equiv (\beta \land \alpha)$ commutativity of $\land$
- $(\alpha \lor \beta) \equiv (\beta \lor \alpha)$ commutativity of $\lor$
- $((\alpha \land \beta) \land \gamma) \equiv (\alpha \land (\beta \land \gamma))$ associativity of $\land$
- $((\alpha \lor \beta) \lor \gamma) \equiv (\alpha \lor (\beta \lor \gamma))$ associativity of $\lor$
- $\neg(\neg \alpha) = \alpha$ double-negation
- $(\alpha \Rightarrow \beta) \equiv (\neg\beta \Rightarrow \neg\alpha)$ contraposition
- $(\alpha \Rightarrow \beta) \equiv (\neg\alpha \lor \beta)$ implication elimination
- $(\alpha \Leftrightarrow \beta) \equiv ((\alpha \Rightarrow \beta) \land (\beta \Rightarrow \alpha))$ biconditional elimination

Logic Equivalence

- $\neg(\alpha \land \beta) \equiv (\neg\alpha \lor \neg\beta)$ De Morgan
- $\neg(\alpha \lor \beta) \equiv (\neg\alpha \land \neg\beta)$ De Morgan
- $(\alpha \land (\beta \lor \gamma)) \equiv ((\alpha \land \beta) \lor (\alpha \land \gamma))$ distributivity of $\land$ over $\lor$
- $(\alpha \lor (\beta \land \gamma)) \equiv ((\alpha \lor \beta) \land (\alpha \lor \gamma))$ distributivity of $\lor$ over $\land$

These equivalences play much the same role in logic as arithmetic identities do in ordinary mathematics.

Validity

- Some sentences are very true! For example

1) True   2) $P \Rightarrow P$ 3) $(P \wedge Q) \Rightarrow Q$

A valid sentence is one whose meaning includes every possible interpretation.

$$((P \vee H) \wedge (\neg H)) \Rightarrow P$$

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

The truth table shows that $((P \vee H) \wedge (\neg H)) \Rightarrow P$ is valid

We write $\models ((P \vee H) \wedge (\neg H)) \Rightarrow P$

Satisfiability

- An unsatisfiable sentence is one whose meaning has no interpretation (e.g., $P \wedge \neg P$)
- A satisfiable sentence is one whose meaning has at least one interpretation.
- A sentence must be either satisfiable or unsatisfiable but it can’t be both.
- If a sentence is valid then it’s satisfiable.
- If a sentence is satisfiable then it may or may not be valid.

Satisfiability

- The SAT problem is to determine the satisfiability of sentences

- Conection to validity:
  - α is valid iff $\neg\alpha$ is unsatisfiable
  - α is satisfiable iff $\neg\alpha$ is not valid

- Proving by checking the unsatisfiability:

  $\alpha \vDash \beta$ if and only if the sentence $(\alpha \land \neg\beta)$ is unsatisfiable

Knowledge Base and Models

- Knowledge base: a set of sentences. Each sentence represents some assertation about the world.
- A model of a set of sentences (KB) is a truth assignment in which each of the KB sentences evaluates to True.
- With more and more sentences, the models of KB start looking more and more like the "real-world".

Models

If a sentence α holds (is True) in all models of a KB, we say that α is entailed by the KB.

α is of interest, because whenever KB is true in a world α will also be True.

We write $KB \models \alpha$

Entailment Examples

KB

- R1: CS4365Lectures ⇒ (TodayIsTuesday V TodayIsThursday)
- R2: ¬TodayIsThursday
- R3: TodayIsSaturday ⇒ SleepLate
- R4: Rainy ⇒ GrassIsWet
- R5: CS4365Lectures V TodayIsSaturday
- R6: ¬ SleepLate

Entailment Examples

KB

R1: CS4365Lectures ⇒ (TodayIsTuesday ∨ TodayIsThursday)

R2: ¬TodayIsThursday

R3: TodayIsSaturday ⇒ SleepLate

R4: Rainy ⇒ GrassIsWet

R5: CS4365Lectures ∨ TodayIsSaturday

R6: ¬ SleepLate

Then which of these are correct entailments?

$KB \models \neg\text{SleepLate}$

$KB \models \text{GrassIsWet}$

$KB \models \neg\text{SleepLate} \vee \text{GrassIsWet}$

$KB \models \text{TodayIsTuesday}$

Entailment Examples

- KB
  - Propositional symbols:
    - CS4365Lectures, TodayIsTuesday, TodayIsThursday, TodayIsSaturday, SleepLate, Rainy, GrassIsWet
  - Model checking:
    - Enumerate all the possible models to check if α is true in all models is in all models in which KB is true

Entailment Examples

KB

- R1: CS4365Lectures ⇒ (TodayIsThursday V TodayIsThusday)
- R2: ¬TodayIsTuesday
- R3: TodayIsSaturday ⇒ SleepLate
- R4: Rainy ⇒ GrassIsWet
- R5: CS4365Lectures V TodayIsSaturday
- R6: ¬ SleepLate

CS4365Lectures: T TodayIsThursday: T TodayIsTuesday: F

TodayIsSaturday: F SleepLate: F Rainy: F/T GrassIsWet: T/F

Entailment Examples

- KB is True when
  - CS4365Lectures: T
  - TodayIsThursday: T
  - TodayIsTuesday: F
  - TodayIsSaturday: F
  - SleepLate: F
  - Rainy: F/T
  - GrassIsWet: T/F

- Complexity: $O(2^N)$

$KB \models \neg \text{SleepLate}$ T

$KB \models \neg \text{SleepLate} \vee \text{GrassIsWet}$ T

$KB \models \text{GrassIsWet}$ F

$KB \models \text{TodayIsTuesday}$ F

Logical Inference

- Problem definition:
  - The computer has a knowledge base KB.
  - The user inputs a sentence.
  - The computer tells the user whether the sentence is entailed by the knowledge base.

Humans who are doing proofs almost never use this brute-force approach. Then how to do logical inference efficiently?

Proof Theory

- A set of purely syntactic rules for efficiently determining entailment
- We write: KB $\vdash \alpha$, i.e., α can be deduced from KB or α is provable from KB.

Key property:

Both in propositional and in first-order logic we have a proof theory (“calculus”) such that:

$\models$ and $\vdash$ are equivalent

Proof Theory (cont.)

If KB $\vdash \alpha$ imples KB $\models \alpha$, we say the proof theory is sound

If KB $\models \alpha$ implies KB $\vdash \alpha$, we say the proof theory is complete.

Why so important?

Allow computer to ignore semantics and “just push symbols”!

Example Proof Theory

One rule of inference: Modus Ponens

From α and $\alpha \Rightarrow \beta$ it follows that β.

Semantic soundness can easily be verified (using truth table).

Another rule of inference: **And-Elimination**

From $\alpha \wedge \beta$, it follows that α and β.

Example Proof Theory

Axiom schemas:

(Ax. I) $\alpha \Rightarrow (\beta \Rightarrow \alpha)$

(Ax. II) $((\alpha \Rightarrow (\beta \Rightarrow \gamma)) \Rightarrow ((\alpha \Rightarrow \beta) \Rightarrow ((\alpha \Rightarrow \gamma)))$

(Ax. III) $(\neg\alpha \Rightarrow \beta) \Rightarrow ((\neg\alpha \Rightarrow \neg\beta) \Rightarrow \alpha)$

Note: α, β, γ stand for arbitrary sentences. So, we have an infinite collection of axioms.

Example Proof

- Now, α can be **deduced** from a set of sentences φ iff there exists a sequence of applications of modus ponens that leads from φ to α (possibly using axioms).
- One can prove that:
  - Modus ponens with the above axioms will generate exactly all (and only those) statements logically entailed by φ.

So, we have a way of generating entailed statements in a purely syntactic manner!

(Sequence is called a proof. Finding it can be hard …)

Example Proof

Lemma. 1) For any α, we have $\vdash (\alpha \Rightarrow \alpha)$.

Proof.

$$(\alpha \Rightarrow ((\alpha \Rightarrow \alpha) \Rightarrow \alpha)) \Rightarrow ((\alpha \Rightarrow (\alpha \Rightarrow \alpha)) \Rightarrow (\alpha \Rightarrow \alpha)), \text{Ax. II}$$

$$\alpha \Rightarrow ((\alpha \Rightarrow \alpha) \Rightarrow \alpha), \text{Ax. I}$$

$$(\alpha \Rightarrow (\alpha \Rightarrow \alpha)) \Rightarrow (\alpha \Rightarrow \alpha); \text{Modus Ponens}$$

$$\alpha \Rightarrow (\alpha \Rightarrow \alpha), \text{Ax. I}$$

$$\alpha \Rightarrow \alpha, \text{Modus Ponens}$$

Another Example Proof

Lemma. 2) For any α and β, we have $\beta, \neg\beta \vdash \alpha$

Proof.

$(\neg\alpha \Rightarrow \beta) \Rightarrow ((\neg\alpha \Rightarrow \neg\beta) \Rightarrow \alpha)$, (Ax. III)

β, (hyp.)

$\beta \Rightarrow (\neg\alpha \Rightarrow \beta)$, (Ax. I)

$\neg\alpha \Rightarrow \beta$, (Modus Ponens)

$(\neg\alpha \Rightarrow \neg\beta) \Rightarrow \alpha$, (Modus Ponens)

$\neg\beta$, (hyp.)

$\neg\beta \Rightarrow (\neg\alpha \Rightarrow \neg\beta)$, (Ax. I)

$\neg\alpha \Rightarrow \neg\beta$, (Modus Ponens)

α, (Modus Ponens)

Another Example Proof

Why are lemma 1 and lemma 2 true semantically?

I.e., $\models \alpha \Rightarrow \alpha$ and $\beta, \neg\beta \models \alpha$

Note: proofs are purely syntactic --- machines does not need to know anything about the meaning of the sentences!

Whatever is **syntactically** derived will be **semantically** true, and we can derive everything syntactically that is semantically true.

How hard is it to find proofs?

Monotonicity

- The set of entailed sentences can only increase as information is added to the knowledge base.

- For any sentence α and β

  if $KB \models \alpha$ then $KB \wedge \beta \models \alpha$

- Propositional logic is monotonic

Key Properties

We have the following properties (also for first-order logic):

For a sound and complete proof theory, the following three conditions are equivalent:

(I) $\varphi \vDash \alpha$

(II) $\varphi \vdash \alpha$

(III) $\varphi, \neg\alpha$ is inconsistent (i.e., can be refuted)

(I) is semantic; (II) syntactic; (III) at high-level semantic but we have a nice syntactic automatic procedure: resolution.

What common proof technique does III represent?