LOGIC:

Knowledge representation

Propositional Logic: ch.7

First-Order-Logic: ch.8

Knowledge Representation (KR)

- Core problem in developing an intelligent system:
  - how express knowledge in a computer-tractable form
- KR: a description that incorporates information about a problem, environment, entity, ...
- Primary focus of KR is two fold; -
  - How to represent the knowledge one has about a problem domain
  - How to reason using that knowledge in order to answer questions or make decisions
- KR deals with the problem of how to model the world sufficiently for intelligent action

How essential is KR ?

- A ‘problem’ involves relationships between concrete objects, abstract concepts
  - relationship: dependencies, constraints, independencies,...
  - an appropriate KR makes these explicit, and clarifies them so as to model them succinctly and without unnecessary details
  - once the KR is designed, then the essence of the problem is clear
  - good representation is key to good problem solving
  - once an appropriate KR is arrived at for a given problem, the problem is almost solved

Evaluating KR

- How do you know that a particular KR is good?
  - Explicitness:clarity is important in expressing state of problem at a glance
  - constraints: expressing how objects, relations influence each other
  - suppress irrelevant detail
  - transparency: easy to understand
  - completeness: all problem variations can be handled
  - concise: compact and clear
  - fast: quick access for reads, writes, updates
  - computable: their creation can be automated
- A problem can be represented in more than one way
  - which is preferrable depends on the goals for the problem solving task and above goals

Knowledge bases

Inference engine	← domain-independent algorithms
Knowledge base	← domain-specific content

Knowledge base (KB) = set of **sentences** in a **knowledge representation language**

Declarative approach to building an agent (or other system):
Tell it what it needs to know

Then it can ASK itself what to do – answers should follow from KB

Agents can be viewed at the **knowledge level**
i.e., what they know, regardless of how implemented
Or at the **implementation level**
i.e., data structures in KB and algorithms that manipulate them

Logic for Knowledge Representation and reasoning

Knowledge-based intelligent agent, one needs:

- to represent the knowledge about the world in a ***formal language***
- to ***reason*** about the world using ***inferences*** in the language
- to decide what action to take by ***inferring*** that the selected action is good

Logic is one of the oldest representation languages studied for AI, and is the foundation for many existing systems that use logic as either inspiration or the basis for the tools in that system, e.g.

- **rule-based expert systems**
- **Prolog programming language**

Logic in general

Logics **are *formal languages* for representing information**

- Such that conclusions can be drawn

syntax: **defines the sentences in the language**

semantics: **defines the "meaning" of sentences.**

- defines *truth* of a sentence in a world

E.g., the language of arithmetic

x + y >= y is a sentence; x2 +y > is not a sentence

x+2 >= y true iff the number x+2 is no less than the number y

x+2>=y is true in a world where x=7, y=1

x+2>=y is false in a world where x=0, y=6

inference procedure: mechanical method for computing (deriving)
New (true) sentences from existing sentences

Types of Logic

Logics –characterized by what they commit to as "primitives"

Ontological Commitment: what exists-facts? Objectives? Objects? Time? Beliefs?
Lepistemological Commitment: what states of knowledge?

Language	Ontological Commitment	Epistemological commitment
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief 0….1
Fuzzy logic	degree of truth	degree of belief 0…..1

Facts: claims about the world that are True or False, whereas
Representation: is an expression (sentence) in some language that can be encoded in a computer program and stands for the objects and relations

Entailment

text, pp. 201-202

Inference

text, pp. 208-210

Propositional logic (PL)

Propositional **(Boolean)** logic – a simple language useful to illustrate basic ideas and definitions

User defines a set of propositional symbols, like P1 and P2
User defines the semantics of these symbols, for example,

P1 means "*its raining*"
P2 means "*it is hot*"

Propositional logic: Syntax

Alphabets of PL contains:

- constants **True** and **False**
- set of propositional symbols (variables) e.g., P and Q
- set of connectives

$$\neg, \vee, \wedge, \rightarrow, \leftrightarrow$$

- the constants are (atomic) sentences by themselves
- propositional symbol P or Q are (atomic)
- if S is a sentence, is $\neg$ (S) a sentence, if S1 is a sentence and S2, then (complex) sentences are formed using logical connective as follows:

(S1 $\wedge$ S2) is a sentence …(and) **conjuction**

(S1 $\vee$ S2) is a sentence ----(or) **disjunction**

(S1 $\Rightarrow$ S2) is a sentence …..**implication**

(S1 $\Leftrightarrow$ S2) is a sentence ….**biconditional**

Propositional logic: Syntax

-precedence (in the absence of parentheses) of the connectives is as follows: $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$

-Propositional variables represent propositions and connectives represent ways of combining the propositions

Example:

P represents the proposition "the lights are on"

Q represents the proposition " the house is locked"

then $P \wedge Q$ represents the sentence "the lights are on *and* the house is locked"

$\neg$ P represents the lights are *not* on"

Propositional logic: Semantics

-semantics of a language give meaning to its sentence
-each model specifies true (T) or false (F) for each proposition symbol

e.g A	B	C
True	*True*	*False*

Rules for evaluating truth wrt a model m:

$\neg$ S is true iff	S is false
S1 $\wedge$ S2 is true iff	S1 is true **and** S2 is true
S1 $\vee$ S2 is true iff	S1 is true **or** S2 is true
S1 $\Rightarrow$ S2 is true iff	S1 is false **or** S2 is true
i.e., is false iff	S1 is true **and** S2 is false
S1 $\Leftrightarrow$ S2 is true iff	S1 $\Rightarrow$ S2 is true **and** S2 $\Rightarrow$ S1 is true

PL: Truth Tables

Inductive cases of the semantics can be expressed as truth tables

A	$\neg A$
T	F
F	T

A	B	$A \vee B$	$A \wedge B$	$A \rightarrow B$	$A \leftrightarrow B$
T	T	T	T	T	T
T	F	T	F	F	F
F	T	T	F	T	F
F	F	F	F	T	T

Validity and Inference

Truth tables can be used not only to define connectives but also to test valid sentences e.g $((P \vee H) \wedge \neg H) \neg P$

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*

If a sentence is **true** in every row, then the **sentence is valid**

Logical Implication:

-logical implication is not equivalent to causal implication as it is used in natural language, for example:

X represents the sentence "Elephants can talk"
Y represents the sentence " Nathan will get an A in COSC 3P71"
Suppose $X \rightarrow Y$

- **in natural language, existence of talking elephants (not performance in the course) is one condition sufficient for Ally to get an A in 3P71**

In propositional logic however, $X \rightarrow Y$ **is true**

Problem Solving using PL

PL can be used to solve a variety of problems e.g.,

- university housing lotteries to decide which student gets first choice of dormitory rooms e.g for the four students Bob, Lisa, Jim and Mary we try to figure how each are ranked wrt in housing dorm given:

    - Lisa is not next to Bob in ranking
    - Jim is ranked immediately ahead of a biology major
    - Bob is ranked immediately ahead of jim
    - one of the women is a biology major
    - Mary of Lisa is ranked first

PL too weak

Propositional Logic (PL) is not a very expressive language because:

- hard to identify “individuals” e.g., Alice, 3
- can’t directly talk about properties of individuals or relations between individuals, e.g., tall(Alex)
- generalization, patterns, regularities can’t easily be represented e.g., all squares have four sides

Summary

- Logical agents apply inference to a knowledge base to derive new information and make decisions
- Basic concepts of logic:
  - **syntax:** formal structure of sentences
  - **semantics**: truth of sentences wrt models
  - **entailment**: necessary truth of one sentence given another
  - **inference**: deriving sentences from other sentences
  - **soundness**: derivations produce only entailed sentences
  - **completeness:** derivations can produce all entailed sentences

- Truth table method is sound and complete for PL, PL lacks expressive power

First-Order Logic

Chapter 8

Pros and cons of propositional logic

(read text, pp 240-241)

- Propositional logic is declarative
- Propositional logic allows partial/disjunctive/negated information
  - (unlike most data structures and databases)
- Propositional logic is compositional:
  - meaning of $A_{1,1} \wedge B_{1,2}$ is derived from meaning of $A_{1,1}$ and of $B_{1,2}$

- Meaning in propositional logic is context-independent
  - (unlike natural language, where meaning depends on context)
- Propositional logic has very limited expressive power
  - (unlike natural language)
  - E.g., cannot say "pits cause breezes in adjacent squares"
    - except by writing one sentence for each square

First-order logic

- Whereas propositional logic assumes the world contains facts
- first-order logic (like natural language) assumes the world contains
  - Objects: people, houses, numbers, colors, baseball games, wars, …
  - Relations: red, round, prime, brother of, bigger than, part of, comes between, …
  - Functions: father of, best friend, one more than, plus, …

Syntax of FOL: Basic elements

Constants	KingJohn, 2, ...
Predicates	Brother, >, before,...
Functions	Sqrt, LeftLegOf,...
Variables	x, y, a, b,...
Connectives	$\neg$, $\Rightarrow$, $\wedge$, $\vee$, $\Leftrightarrow$
Equality	=
Quantifiers	$\forall$, $\exists$

Atomic sentences

Atomic sentence = *predicate* $(term_1,\ldots,term_n)$
or $term_1 = term_2$

- E.g Married (Father(Richard), Mother(John))

Term = *function* $(term_1,\ldots,term_n)$
or *constant* or *variable*

Complex sentences

- Complex sentences are made from atomic sentences using connectives

$$\neg S,\ S_1 \wedge S_2,\ S_1 \vee S_2,\ S_1 \Rightarrow S_2,\ S_1 \Leftrightarrow S_2,$$

E.g. *Sibling(KingJohn,Richard)* $\Rightarrow$ *Sibling(Richard,KingJohn)*

Truth in first-order logic

- Sentences are true with respect to a model and an interpretation
- Model contains objects (domain elements) and relations among them
- Interpretation specifies referents for

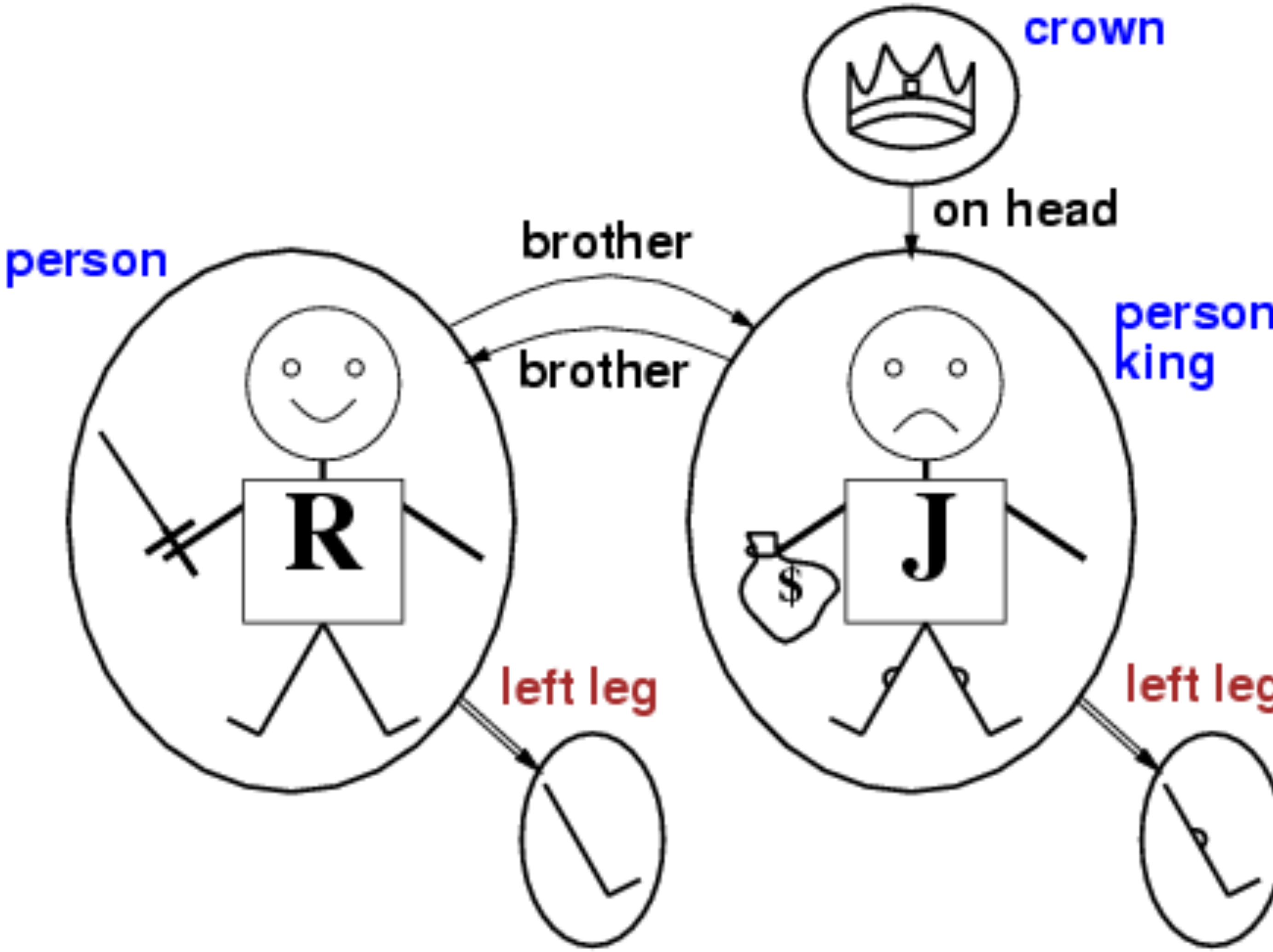
  constant symbols → objects

  predicate symbols → relations

  function symbols → functional relations

- An atomic sentence *predicate*$(term_1,\ldots,term_n)$ is true iff the objects referred to by $term_1,\ldots,term_n$ are in the relation referred to by *predicate*

Models for FOL: Example



Universal quantification

- $\forall$<*variables*> <*sentence*>

Everyone at Brock is smart:
$\forall x\ At(x, Brock) \Rightarrow Smart(x)$

- Generally, equivalent to the conjunction of instantiations

$$
\begin{aligned}
& At(KingJohn, Brock) \Rightarrow Smart(KingJohn) \\
\wedge\ & At(Richard, Brock) \Rightarrow Smart(Richard) \\
\wedge\ & At(Brock, Brock) \Rightarrow Smart(Brock) \\
\wedge\ & \ldots
\end{aligned}
$$

- “ All kings are persons”:

$\forall x\ \ King(x,) \Rightarrow Person(x)$

A common mistake to avoid

- Typically, $\Rightarrow$ is the main connective with $\forall$
- Common mistake: using $\wedge$ as the main connective with $\forall$:

  $\forall x\ At(x,Brock) \wedge Smart(x)$

  means "Everyone is at Brock and everyone is smart"

Existential quantification

- $\exists$<*variables*> <*sentence*>
- Someone at BROCK is smart:
- $\exists x$ At(x, BROCK) $\wedge$ Smart(x)
- Roughly speaking, equivalent to the disjunction of instantiations

  At(KingJohn, BROCK) $\wedge$ Smart(KingJohn)
  $\vee$ At(Richard, BROCK) $\wedge$ Smart(Richard)
  $\vee$ At(BROCK, BROCK) $\wedge$ Smart(BROCK)
  $\vee$ ...

Another common mistake to avoid

- Typically, $\wedge$ is the main connective with $\exists$

- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:

$$\exists x\ At(x, BROCK) \Rightarrow Smart(x)$$

is true if there is anyone who is not at BROCK!

Properties of quantifiers

- $\forall x\ \forall y$ is the same as $\forall y\ \forall x$
- $\exists x\ \exists y$ is the same as $\exists y\ \exists x$

- $\exists x\ \forall y$ is not the same as $\forall y\ \exists x$
- $\exists x\ \forall y\ Loves(x,y)$
  - "There is a person who loves everyone in the world"
- $\forall y\ \exists x\ Loves(x,y)$
  - "Everyone in the world is loved by at least one person"

- Quantifier duality: each can be expressed using the other
- $\forall x\ Likes(x,IceCream) \quad \neg\exists x\ \neg Likes(x,IceCream)$
- $\exists x\ Likes(x,Broccoli) \qquad\qquad \neg\forall x\ \neg Likes(x,Broccoli)$

Equality

- $term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object

- E.g., definition of *Sibling* in terms of *Parent*:
  $\forall x,y\ Sibling(x,y) \Leftrightarrow [\neg(x = y) \wedge \exists m,f\ \neg(m = f) \wedge Parent(m,x) \wedge Parent(f,x) \wedge Parent(m,y) \wedge Parent(f,y)]$

Using FOL

The kinship domain:

- Brothers are siblings

  $\forall x,y \; Brother(x,y) \Leftrightarrow Sibling(x,y)$

- One's mother is one's female parent

  $\forall m,c \; Mother(c) = m \Leftrightarrow (Female(m) \wedge Parent(m,c))$

- "Sibling" is symmetric

  $\forall x,y \; Sibling(x,y) \Leftrightarrow Sibling(y,x)$

Knowledge engineering in FOL

1. Identify the task
2. Assemble the relevant knowledge
3. Decide on a vocabulary of predicates, functions, and constants
4. Encode general knowledge about the domain
5. Encode a description of the specific problem instance
6. Pose queries to the inference procedure and get answers
7. Debug the knowledge base

Summary of FOL

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Increased expressive power