Artificial Intelligence

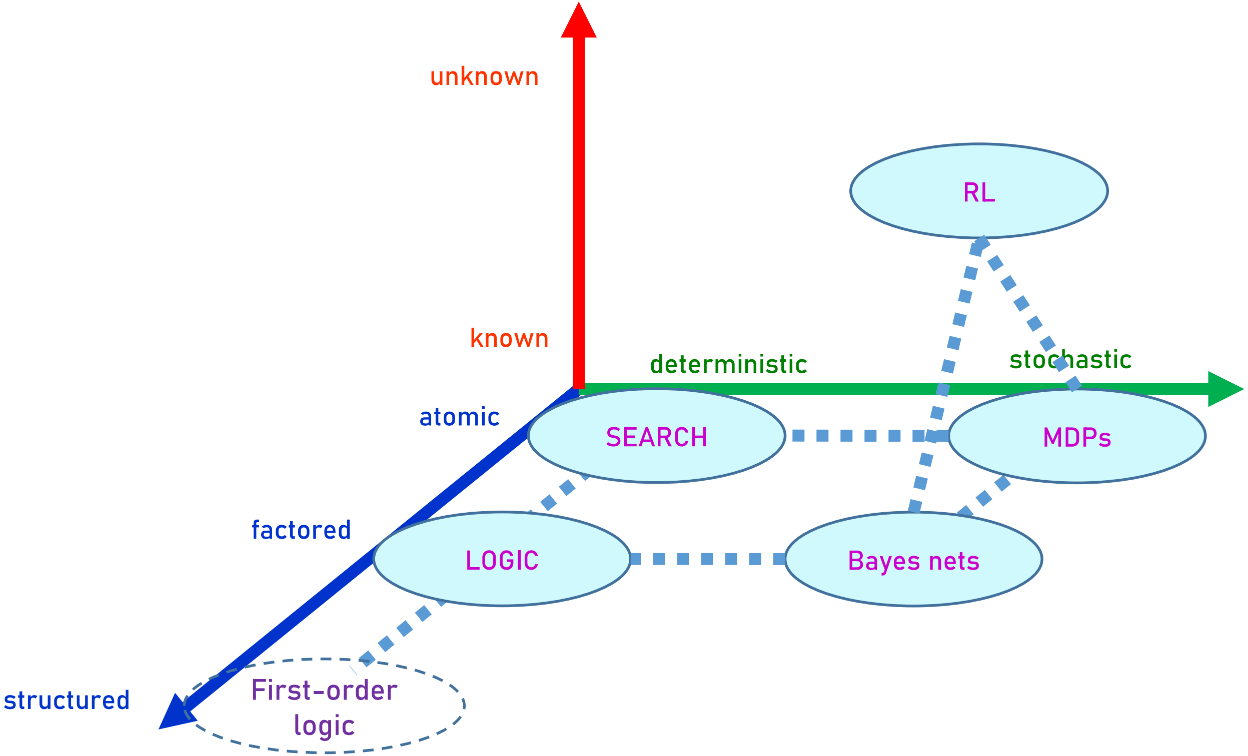
8. First-Order Logic

Shashi Prabh

School of Engineering and Applied Science

Ahmedabad University

Contents



unknown
known
deterministic
stochastic
atomic
factored
structured
RL
SEARCH
MDPs
LOGIC
Bayes nets
First-order
logic

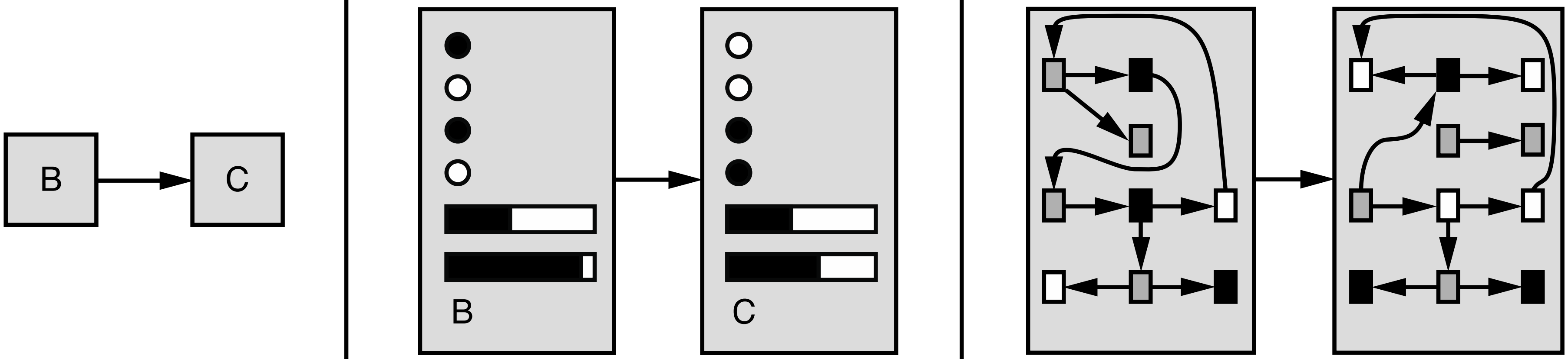
Contents

Goal: Design knowledge-based agents that use reasoning over an internal representation of knowledge to decide the actions.

Topics

- First-Order (or, Predicate) Logic: syntax and semantics
- Inference in First-Order Logic

Spectrum of representations



(a) Atomic | (b) Factored | (b) Structured

Search, game-playing

CSPs, planning, propositional logic, Bayes nets, neural nets

First-order logic, databases, logic programs, probabilistic programs

First-Order Logic (FOL)

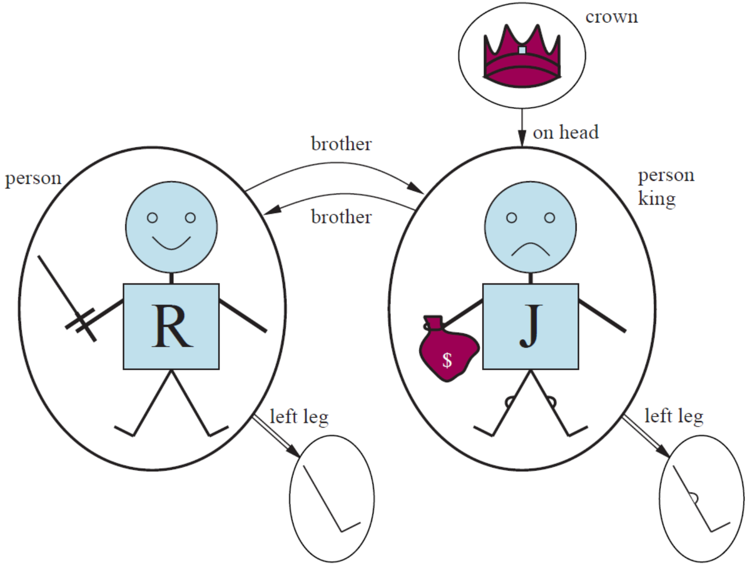
- Also called predicate logic
- Common high level programming languages lack a general mechanism for deriving facts from other facts.
  - Programmers write domain-specific procedures to do so.
- In declarative approach, such as propositional and first-order logic, knowledge and inference are separate
  - Inference is domain independent
- Propositional logic deals with sentences (facts) that are true or false (or unknown)
- FOL deals with sentences plus objects and relations
  - Some relations among the objects may or may not hold

Expressive power of FOL

- Rules of chess:
  - 100,000 pages in propositional logic
  - 1 page in first-order logic
- Saying some fact about 100 people in propositional logic requires 100 sentences
  - 1 sentence in FOL

FOL Syntax

- A FOL model containing five objects
  - R, J, RL, JL, C
- Two binary relations
  - Brotherhood: {<R,J>, <J,R>}
  - On-head: {<C, J>}
- Three unary relations
  - Person: {R, J}
  - King: {J}
  - Crown: {J}
- One unary function (object to object relation)
  - Left-leg: <R> → RL, <J> → JL



Syntax and semantics: Terms

$$
\begin{aligned}
\mathit{Sentence} &\rightarrow \mathit{AtomicSentence} \mid \mathit{ComplexSentence} \\
\mathit{AtomicSentence} &\rightarrow \mathit{Predicate} \mid \mathit{Predicate}(\mathit{Term},\ldots) \mid \mathit{Term} = \mathit{Term} \\
\mathit{ComplexSentence} &\rightarrow (\,\mathit{Sentence}\,) \\
&\mid \neg\, \mathit{Sentence} \\
&\mid \mathit{Sentence} \land \mathit{Sentence} \\
&\mid \mathit{Sentence} \lor \mathit{Sentence} \\
&\mid \mathit{Sentence} \;\Rightarrow\; \mathit{Sentence} \\
&\mid \mathit{Sentence} \;\Leftrightarrow\; \mathit{Sentence} \\
&\mid \mathit{Quantifier}\ \mathit{Variable},\ldots\ \mathit{Sentence} \\
\\
\mathit{Term} &\rightarrow \mathit{Function}(\mathit{Term},\ldots) \\
&\mid \mathit{Constant} \\
&\mid \mathit{Variable} \\
\\
\mathit{Quantifier} &\rightarrow \forall \mid \exists \\
\mathit{Constant} &\rightarrow A \mid X_1 \mid \mathit{John} \mid \cdots \\
\mathit{Variable} &\rightarrow a \mid x \mid s \mid \cdots \\
\mathit{Predicate} &\rightarrow \mathit{True} \mid \mathit{False} \mid \mathit{After} \mid \mathit{Loves} \mid \mathit{Raining} \mid \cdots \\
\mathit{Function} &\rightarrow \mathit{Mother} \mid \mathit{LeftLeg} \mid \cdots \\
\textsc{Operator Precedence} &: \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
\end{aligned}
$$

Syntax and semantics: Terms

- A term refers to an object
- A term can be:
  - A constant symbol
    - 2, A , B, John
    - The possible world fixes these referents
  - A logical variable
    - x, kings
  - A function symbol with terms as arguments
    - Add (x, 5), sqrt(x), Country(kings)

Syntax and semantics: Atomic sentences

- Atomic sentence (or, Atomic Formula) state assertions
  - Brother (R, J)
  - Married (Father(R), Mother(J))
  - LivesIn (x, Delhi))
- Analogous to symbols in PL
- An atomic sentence is True iff the objects referred to by the terms are in the relation referred to by the predicate

Syntax and semantics: Complex sentences

- Using logical connectives to construct more complex sentences
  - $\neg\alpha$, $\alpha \wedge \beta$, $\alpha \vee \beta$, $\alpha \Rightarrow \beta$, $\alpha \Leftrightarrow \beta$
  - Brother (R, J) $\wedge$ Brother (J, R)
  - $\neg$ Brother (Left-leg(R), J)
  - $\neg$King (R) $\Rightarrow$ King (J)
- Sentences with universal or existential quantifiers
  - Universal quantification ($\forall$, forall)
    - $\forall$x King (x) $\Rightarrow$ Person (x)
    - $\forall$x Dog (x) $\Rightarrow$ Eats (x, DogFood)
    - Conjunction: $\forall x\ P(x) \equiv P(x_1) \wedge P(x_2) \wedge \ldots$
  - Existential quantification ($\exists$, there exists)
    - $\exists$x Likes(x, Cheese)
    - Disjunction: $\exists\ x\ P(x) \equiv P(x_1) \vee P(x_2) \vee \ldots$

Fun with sentences

- Everyone knows Potter
  - $\forall n\ Person(n) \Rightarrow Knows(n, Potter)$
- There is someone that everyone knows
  - $\exists s\ Person(s) \land \forall n\ Person(n) \Rightarrow Knows(n,s)$
- Everyone knows someone
  - $\forall x\ Person(x) \Rightarrow \exists y\ Person(y) \land Knows(x,y)$

Nested quantifiers

- Every brother is a sibling
  - $\forall x,y$ Brother (x, y) $\Rightarrow$ Sibling (x, y)
- Order matters
  - Everybody likes somebody: $\forall x\ \exists y$ Likes (x, y)
  - Somebody is liked by everyone: $\exists y\ \forall x$ Likes (x, y)
- Universal and existential quantifiers obey De Morgan's rules
- $\forall x\ \neg$Like(x, Pepper) $\equiv \neg\ \exists$ x Likes (x, Pepper)
  - Nobody likes pepper
- $\forall x$ Like(x, IceCream) $\equiv \neg\ \exists$ x $\neg$ Likes (x, IceCream)
  - Everybody likes Icecream

Inference in FOL

- Entailment is defined exactly as for propositional logic:
  - $\alpha \models \beta$ ("α entails β") iff in every world where α is true, β is also true
  - E.g., $\forall x$ Knows(x, Logic) entails $\exists y \forall x$ Knows(x, y)
- In FOL, we can go beyond just answering "yes" or "no"; given an existentially quantified query, return a substitution (or binding) for the variable(s) such that the resulting sentence is entailed:
  - KB = $\forall x$ Knows(x, Logic)
  - Query = $\exists y\ \forall x$ Knows(x, y)
  - Answer = Yes, σ = {y/Logic}
  - Notation: $\alpha\sigma$ means applying substitution σ to sentence α
    - E.g., if α= $\forall x$ Knows(x,y) and σ = {y/Logic}, then $\alpha\sigma$ = $\forall x$ Knows(x,Logic)

Inference in FOL: Lifted inference

- Apply inference rules directly to first-order sentences, e.g.,
  - KB = Person(Socrates), $\forall$x Person(x) $\Rightarrow$ Mortal(x)
  - conclude Mortal(Socrates)
  - The general rule is a version of Modus Ponens:
    - Given $\alpha \Rightarrow \beta$ and α', where $\alpha'\sigma = \alpha\sigma$ for some substitution σ, conclude $\beta\sigma$
      - σ is {x/Socrates}
    - Given Knows(x, Logic) and Knows(y,z) $\Rightarrow$ Likes(y,z)
      - σ is {y/x, z/Logic}, conclude Likes(x, Logic)
- Examples: Prolog (backward chaining), Datalog (forward chaining), production rule systems (forward chaining), resolution theorem provers

Summary

- FOL is a very expressive formal language
- Many domains of common-sense and technical knowledge can be written in FOL
  - circuits, software, planning, law, taxes, network and security protocols, product descriptions, ecommerce transactions, geographical information systems, Google Knowledge Graph, Semantic Web, etc.
- Inference is semidecidable in general but many problems are efficiently solvable in practice

Next Week

- Probability Review
- Bayesian Networks
  - Inference in Bayes nets