Logic

CPSC 447 - Artificial Intelligence I

General AI Process

- data $\rightarrow$ learning $\rightarrow$ model $\rightarrow$ inference
- question $\rightarrow$ inference $\rightarrow$ answer
- Examples: search problems, games, constraint satisfaction problems (CSP), Markov decision processes (MDP), Bayesian networks

Modeling Paradigms

- State-based models: search problems, games, MDPs
  - 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

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
  - Example: $\mathit{Rain} \wedge \mathit{Wet}$
- Semantics: for each formula f, specify a set of models $M(f)$
  - Example:

Rain	Wet
T	T
T	F
F	T
F	F

- Inference rules: given KB, what new formulas f can be derived?
  - Example: from $\mathit{Rain} \wedge \mathit{Wet}$, derive Rain

Inference Algorithm

- Inference algorithm: repeatedly apply inference rules to derive new formulas.
- Desiderata: soundness and completeness
  - entailment ($KB \models f$)
  - derivation ($KB \vdash f$)

Formulas

- Propositional logic: any legal combination of symbols

$$(\mathit{Rain} \wedge \mathit{Snow}) \rightarrow (\mathit{Traffic} \vee \mathit{Peaceful}) \wedge \mathit{Wet}$$

- Propositional logic with only Horn clauses (restricted)

$$(\mathit{Rain} \wedge \mathit{Snow}) \rightarrow \mathit{Traffic}$$

Tradeoffs

Formulas allowed	Inference rule	Complete?
Propositional logic	modus ponens	no
Horn clauses	modus ponens	yes
Propositional logic	resolution	yes

Resolution Algorithm

- Relationship between entailment and contradiction
- Algorithm: resolution-based inerence
  1. Add $\neg f$ to KB
  2. Convert all formulas to conjunctive normal form (CNF)
  3. Repeatedly apply resolution rule
  4. Return entailment iff derive false

Modus Ponens versus Resolution

	Horn clauses	Any clauses
Inference rule	modus ponens	resolution
Complexity	linear time	exponential time
Expressiveness	less expressive	more expressive

Syntax of First-Order Logic (FOL)

- Terms (refer to objects):
    - Constant symbol
    - Variable
    - Function of terms
- Formulas (refer to truth values)
    - Atomic formulas (atoms)
    - Connectives applied to formulas
    - Quantifiers applied to formulas

Models in First-Order Logic

- A model represents a possible situation in the world.
- A model w in propositional logic maps propositional symbols to truth values
- A model w in first-order logic maps to:
  - constant symbols to objects
  - predicate symbols to tuples of objects

A Restriction on Models

- Unique names assumption: each object has at most one constant symbol.
- Domain closure: each object has at least one constant symbol
- That is, constant symbol $\leftrightarrow$ object

Propositionalization

- If one-to-one mapping between constant symbols and objects (unique names and domain closure), then FOL is syntactic sugar for propositional logic
- Example FOL knowledge base
  - $\mathit{Student}(\mathit{alice}) \wedge \mathit{Student}(\mathit{bob})$
  - $\forall x\ \mathit{Student}(x) \rightarrow \mathit{Person}(x)$
  - $\exists x\ \mathit{Student}(x) \wedge \mathit{Creative}(x)$
- Example propositional logic knowledge bas:
  - $\mathit{StudentAlice} \wedge \mathit{StudentBob}$
  - $(\mathit{StudentAlice} \rightarrow \mathit{PersonAlice}) \wedge (\mathit{StudentBob} \rightarrow \mathit{PersonBob})$
  - $(\mathit{StudentAlice} \wedge \mathit{CreativeAlice}) \vee (\mathit{StudentBob} \wedge \mathit{CreativeBob})$
- Point: use any inference algorithm for propositional logic

Modus Ponens

- Given $P(alice)$ and $\forall x\ P(x) \rightarrow Q(x)$
- Problem: cannot infer $Q(alice)$ because $P(x)$ and $P(alice)$ do not match
- Solution: substitution and unification

Substitution

- Definition: A substitution θ is a mapping from variables to terms. $\mathit{Subst}[\theta, f]$ denotes the result of performing substitution θ on f
- Example:

$$\mathit{Subst}[\{x/\mathit{alice}\}, P(x)] = P(\mathit{alice})$$

Unification

- Definition: Unification takes two formulas f and g and returns a substitution θ with is the most general unifier: $\mathit{Unify}[f, g] = \theta$ such that $\mathit{Subst}[\theta, f] = \mathit{Subst}[\theta, g]$ or "fail" if no such θ exists.
- Example:

$$\mathit{Unify}[\mathit{Knows}(\mathit{alice}, \mathit{arithmetic}), \mathit{Knows}(x, \mathit{arithmetic})] = \{x/\mathit{alice}\}$$

Modus Ponens (FOL)

$$\frac{a'_1, \ldots a'_k,\ \forall x_1 \ldots, \forall x_n (a_1 \wedge \ldots \wedge a_k) \rightarrow b}{b'}$$

- Get the most general unifier θ on premises:

$$\theta = \mathit{Unify}[a'_1 \wedge \ldots \wedge a'_k, a_1 \wedge \ldots \wedge a_k]$$

- Apply θ to conclusion:

$$\mathit{Subst}[\theta, b] = b'$$