Artificial Intelligence

Propositional Logic
(Chapter 7)

Instructor: Qiang Yu

Logic and AI

- Would like our AI to have knowledge about the world, and logically draw conclusions from it
- Search algorithms generate successors and evaluate them, but do not "understand" much about the setting
- Example question: is it possible for a chess player to have 8 pawns and 2 queens?
  - Search algorithm could search through tons of states to see if this ever happens, but…

A story

- Your roommate comes home; he/she is completely wet
- You know the following things:
  - Your roommate is wet
  - If your roommate is wet, it is because of rain, sprinklers, or both
  - If your roommate is wet because of sprinklers, the sprinklers must be on
  - If your roommate is wet because of rain, your roommate must not be carrying the umbrella
  - The umbrella is not in the umbrella holder
  - If the umbrella is not in the umbrella holder, either you must be carrying the umbrella, or your roommate must be carrying the umbrella
  - You are not carrying the umbrella
- Can you conclude that the sprinklers are on?
- Can AI conclude that the sprinklers are on?

Knowledge base for the story

- RoommateWet
- RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)
- RoommateWetBecauseOfSprinklers => SprinklersOn
- RoommateWetBecauseOfRain => NOT(RoommateCarryingUmbrella)
- UmbrellaGone
- UmbrellaGone => (YouCarryingUmbrella OR RoommateCarryingUmbrella)
- NOT(YouCarryingUmbrella)

Syntax

- What do well-formed sentences in the knowledge base look like?
- A BNF grammar:
- *Symbol* → P, Q, R, …, RoommateWet, …
- *Sentence* → True | False | *Symbol* | NOT(*Sentence*) | (*Sentence* AND *Sentence*) | (*Sentence* OR *Sentence*) | (*Sentence* => *Sentence*)
- We will drop parentheses sometimes, but formally they really should always be there

Semantics

- A model specifies which of the proposition symbols are true and which are false
- Given a model, I should be able to tell you whether a sentence is true or false

$\neg, \wedge, \vee, \Rightarrow,$

- Truth table defines semantics of operators:

a	b	NOT(a)	a AND b	a OR b	a => b
false	false	true	false	false	true
false	true	true	false	true	true
true	false	false	false	true	false
true	true	false	true	true	true

- Given a model, can compute truth of sentence recursively with these

Caveats

- TwoIsAnEvenNumber OR ThreeIsAnOddNumber

  is true (not exclusive OR)

- TwoIsAnOddNumber => ThreeIsAnEvenNumber

  is true (if the left side is false it's always true)

All of this is assuming those symbols are assigned their natural values…

Tautologies

- A sentence is a tautology if it is true for any setting of its propositional symbols

P	Q	P OR Q	NOT(P) AND NOT(Q)	(P OR Q) OR (NOT(P) AND NOT(Q))
false	false	false	true	true
false	true	true	false	true
true	false	true	false	true
true	true	true	false	true

- (P OR Q) OR (NOT(P) AND NOT(Q)) is a tautology

Logical equivalences

- Two sentences are logically equivalent if they have the same truth value for every setting of their propositional variables

P	Q	P OR Q	NOT(NOT(P) AND NOT(Q))
false	false	false	false
false	true	true	true
true	false	true	true
true	true	true	true

- P OR Q and NOT(NOT(P) AND NOT(Q)) are logically equivalent

Famous logical equivalences

- (a OR b) ≡ (b OR a) *commutatitvity*
- (a AND b) ≡ (b AND a) *commutatitvity*
- ((a AND b) AND c) ≡ (a AND (b AND c)) *associativity*
- ((a OR b) OR c) ≡ (a OR (b OR c)) *associativity*
- NOT(NOT(a)) ≡ a *double-negation elimination*
- (a => b) ≡ (NOT(b) => NOT(a)) *contraposition*
- (a => b) ≡ (NOT(a) OR b) *implication elimination*
- NOT(a AND b) ≡ (NOT(a) OR NOT(b)) *De Morgan*
- NOT(a OR b) ≡ (NOT(a) AND NOT(b)) *De Morgan*
- (a AND (b OR c)) ≡ ((a AND b) OR (a AND c)) *distributitivity*
- (a OR (b AND c)) ≡ ((a OR b) AND (a OR c)) *distributitivity*

Inference

- We have a knowledge base of things that we know are true
  - RoommateWetBecauseOfSprinklers
  - RoommateWetBecauseOfSprinklers => SprinklersOn
- Can we conclude that SprinklersOn?
- We say SprinklersOn is entailed by the knowledge base if, for every setting of the propositional variables for which the knowledge base is true, SprinklersOn is also true

RWBOS	SprinklersOn	Knowledge base
false	false	false
false	true	false
true	false	false
true	true	true

- SprinklersOn is entailed!

Simple algorithm for inference

- Want to find out if sentence a is entailed by knowledge base…
- *For every possible setting of the propositional variables,*
  - *If knowledge base is true and a is false, return false*
- *Return true*
- Not very efficient: $2^{\text{\#propositional variables}}$ settings

First-Order Logic
(Chapter 8)

Limitations of propositional logic

- So far we studied propositional logic
- Some English statements are hard to model in propositional logic:
- "If your roommate is wet because of rain, your roommate must not be carrying **any** umbrella"
- Pathetic attempt at modeling this:
- RoommateWetBecauseOfRain => (NOT(RoommateCarryingUmbrella0) AND NOT(RoommateCarryingUmbrella1) AND NOT(RoommateCarryingUmbrella2) AND …)

Problems with propositional logic

- No notion of objects
- No notion of relations among objects
- RoommateCarryingUmbrella0 is instructive **to us**, suggesting
  - there is an object we call Roommate,
  - there is an object we call Umbrella0,
  - there is a relationship Carrying between these two objects
- Formally, none of this meaning is there
  - Might as well have replaced RoommateCarryingUmbrella0 by P

Elements of first-order logic

- Objects: can give these names such as Umbrella0, Person0, John, Earth, …
- Relations: Carrying(., .), IsAnUmbrella(.)
  - Carrying(Person0, Umbrella0), IsUmbrella(Umbrella0)
  - Relations with one object = unary relations = properties
- Functions: Roommate(.)
  - Roommate(Person0)
- Equality: Roommate(Person0) = Person1

Things to note about functions

- It could be that we have a separate name for Roommate(Person0)
- E.g., Roommate(Person0) = Person1
- … but we do not **need** to have such a name

- A function can be applied to any object
- E.g., Roommate(Umbrella0)

Reasoning about many objects at once

- Variables: x, y, z, … can refer to multiple objects
- New operators “for all” and “there exists”
  - Universal quantifier and existential quantifier

$\forall$

$\exists$

- for all x: CompletelyWhite(x) => NOT(PartiallyBlack(x))
  - Completely white objects are never partially black
- there exists x: PartiallyWhite(x) AND PartiallyBlack(x)
  - There exists some object in the world that is partially white and partially black

Practice converting English to first-order logic

- “John has Jane’s umbrella”
- Has(John, Umbrella(Jane))
- “John has an umbrella”
- there exists y: (Has(John, y) AND IsUmbrella(y))
- “Anything that has an umbrella is not wet”
- for all x: ((there exists y: (Has(x, y) AND IsUmbrella(y))) => NOT(IsWet(x)))
- “Any person who has an umbrella is not wet”
- for all x: (IsPerson(x) => ((there exists y: (Has(x, y) AND IsUmbrella(y))) => NOT(IsWet(x))))

Axioms and theorems

- Axioms: basic facts about the domain, our “initial” knowledge base
- Theorems: statements that are logically derived from axioms