Dr. George Karraz, Ph. D.

Artificial Intelligence
Lecture II

Propositional Logic

Dr. George Karraz, Ph. D.

Contents:

1. Introduction to Logic (Logic & AI)
2. Knowledge Base
    - Syntax
    - Semantics
    - Logical equivalences
    - Inference
    - Inconsistent knowledge bases
3. Reasoning patterns
4. Unit resolution

1. Introduction to Logic (Logic & AI), cont..

A story

- You 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?

2. Knowledge base

for the story

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

2. Knowledge base, cont…

- Syntax

- What do well-formed sentences in the knowledge base look like?
- A 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

2. Knowledge base, cont…

• 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
- 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)

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

Is this a tautology?

- (P => Q) OR (Q => P)

2. Knowledge base, cont…

• 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
- Tautology = logically equivalent to True

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)) *distributivity*
- (a OR (b AND c)) ≡ ((a OR b) AND (a OR c)) *distributivity*

2. Knowledge base, cont…

• 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	true
true	false	false
true	true	true

2. Knowledge base, cont…

- Inconsistent knowledge bases

- Suppose we were careless in how we specified our knowledge base:
- PetOfRoommateIsABird => PetOfRoommateCanFly
- PetOfRoommateIsAPenguin => PetOfRoommateIsABird
- PetOfRoommateIsAPenguin => NOT(PetOfRoommateCanFly)
- PetOfRoommateIsAPenguin
- No setting of the propositional variables makes all of these true
- Therefore, technically, this knowledge base implies anything

3. Reasoning patterns

- Obtain new sentences directly from some other sentences in knowledge base according to reasoning patterns
- If we have sentences a and a => b , we can correctly conclude the new sentence b
  - This is called modus ponens
- If we have a AND b , we can correctly conclude a
- All of the logical equivalences from before also give reasoning patterns

3. Reasoning patterns, cont…

Formal proof that the sprinklers are on

1) RoommateWet
2) RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)
3) RoommateWetBecauseOfSprinklers => SprinklersOn
4) RoommateWetBecauseOfRain => NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) UmbrellaGone => (YouCarryingUmbrella OR RoommateCarryingUmbrella)
7) NOT(YouCarryingUmbrella)
8) YouCarryingUmbrella OR RoommateCarryingUmbrella *(modus ponens on 5 and 6)*
9) NOT(YouCarryingUmbrella) => RoommateCarryingUmbrella *(equivalent to 8)*
10) RoommateCarryingUmbrella *(modus ponens on 7 and 9)*
11) NOT(NOT(RoommateCarryingUmbrella) *(equivalent to 10)*
12) NOT(NOT(RoommateCarryingUmbrella)) => NOT(RoommateWetBecauseOfRain) *(equivalent to 4 by contraposition)*
13) NOT(RoommateWetBecauseOfRain) *(modus ponens on 11 and 12)*
14) RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers *(modus ponens on 1 and 2)*
15) NOT(RoommateWetBecauseOfRain) => RoommateWetBecauseOfSprinklers *(equivalent to 14)*
16) RoommateWetBecauseOfSprinklers *(modus ponens on 13 and 15)*
17) SprinklersOn *(modus ponens on 16 and 3)*

3. Reasoning patterns, cont..

Reasoning about penguins

1) PetOfRoommateIsABird => PetOfRoommateCanFly
2) PetOfRoommateIsAPenguin => PetOfRoommateIsABird
3) PetOfRoommateIsAPenguin => NOT(PetOfRoommateCanFly)
4) PetOfRoommateIsAPenguin
5) PetOfRoommateIsABird *(modus ponens on 4 and 2)*
6) PetOfRoommateCanFly *(modus ponens on 5 and 1)*
7) NOT(PetOfRoommateCanFly) *(modus ponens on 4 and 3)*
8) NOT(PetOfRoommateCanFly) => FALSE *(equivalent to 6)*

3. Reasoning patterns, cont..

Getting more systematic

- Any knowledge base can be written as a single formula in conjunctive normal form (CNF)
  - CNF formula: (… OR … OR …) AND (… OR …) AND …
  - … can be a symbol x, or NOT(x)
  - Multiple facts in knowledge base are effectively ANDed together

RoommateWet => (RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)

becomes

(NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers)

3. Reasoning patterns, cont…

Converting story problem to conjunctive normal form

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

4. Unit resolution

- Unit resolution: if we have
- l_1 OR l_2 OR … OR l_k

  and
- NOT(l_i)

  we can conclude

  $$l_1 \text{ OR } l_2 \text{ OR } \dots\ l_{i-1} \text{ OR } l_{i+1} \text{ OR } \dots \text{ OR } l_k$$
- Basically modus ponens

4. Unit resolution, cont…

- Applying resolution to story problem

1) RoommateWet
2) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers
3) NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn
4) NOT(RoommateWetBecauseOfRain) OR NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) NOT(UmbrellaGone) OR YouCarryingUmbrella OR RoommateCarryingUmbrella
7) NOT(YouCarryingUmbrella)
8) NOT(UmbrellaGone) OR RoommateCarryingUmbrella (6,7)
9) RoommateCarryingUmbrella (5,8)
10) NOT(RoommateWetBecauseOfRain) (4,9)
11) NOT(RoommateWet) OR RoommateWetBecauseOfSprinklers (2,10)
12) RoommateWetBecauseOfSprinklers (1,11)
13) SprinklersOn (3,12)

4. Unit resolution, cont…

(General) resolution

- General resolution: if we have
- l_1 OR l_2 OR … OR l_k

  and
- m_1 OR m_2 OR … OR m_n

  where for some i,j, $l_i = \text{NOT}(m_j)$

  we can conclude
- l_1 OR l_2 OR … l_{i-1} OR l_{i+1} OR … OR l_k OR m_1 OR m_2 OR … OR m_{j-1} OR m_{j+1} OR … OR m_n
- Same literal may appear multiple times; remove those

4. Unit resolution, cont…

- Applying resolution to story problem (more clumsily)

1) RoommateWet
2) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR RoommateWetBecauseOfSprinklers
3) NOT(RoommateWetBecauseOfSprinklers) OR SprinklersOn
4) NOT(RoommateWetBecauseOfRain) OR NOT(RoommateCarryingUmbrella)
5) UmbrellaGone
6) NOT(UmbrellaGone) OR YouCarryingUmbrella OR RoommateCarryingUmbrella
7) NOT(YouCarryingUmbrella)
8) NOT(RoommateWet) OR RoommateWetBecauseOfRain OR SprinklersOn (2,3)
9) NOT(RoommateCarryingUmbrella) OR NOT(RoommateWet) OR SprinklersOn (4,8)
10) NOT(UmbrellaGone) OR YouCarryingUmbrella OR NOT(RoommateWet) OR SprinklersOn (6,9)
11) YouCarryingUmbrella OR NOT(RoommateWet) OR SprinklersOn (5,10)
12) NOT(RoommateWet) OR SprinklersOn (7,11)
13) SprinklersOn (1,12)