NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CZ3005 Artificial Intelligence

Week 9a – Propositional Logic

Yu Han

han.yu@ntu.edu.sg

Nanyang Assistant Professor
School of Computer Science and Engineering
Nanyang Technological University

Logic is a Formal Language

- **Propositions:**
  – Anil is Intelligent
  – Anil is hardworking
  – If Anil is Intelligent and Anil is Hardworking, then Anil scores a high mark

Propositional Logic

- **Syntax** of the representation language specifies all the sentences that are well-formed.

- **Semantics** of the language defines the truth of each sentence with respect to each possible world.

Elements of Propositional Logic

- **Symbols**
  – Logical constants: TRUE, FALSE
  – Propositional symbols: P, Q, etc. (uppercase)
  – Logical connectives: $\wedge$, $\vee$, $\Leftrightarrow$, $\Rightarrow$, $\neg$
  – Parentheses: ( )
- **Sentences**
  – Atomic sentences: constants, propositional symbols
  – Combined with connectives, e.g. $P \wedge Q \vee R$
  also wrapped in parentheses, e.g. $(P \wedge Q) \vee R$

Elements of Propositional Logic

- Anil is intelligent = Intelligent(Anil)
- Anil is hardworking = Hardworking(Anil)

Propositions

- Objects and relations or Functions

Anil   hardworking, intelligent

- A proposition (statement) can be true or false

Logical Connectives

- **Conjunction** $\wedge$
  - Binary op., e.g. P $\wedge$ Q, “P and Q”, where P, Q are the *conjuncts*
- **Disjunction** $\vee$
  - Binary op., e.g. P $\vee$ Q, “P or Q”, where P, Q are the *disjuncts*
- **Implication** $\Rightarrow$
  - Binary op., e.g. P $\Rightarrow$ Q, “P implies Q”, where P is the *premise* (antecedent) and Q the *conclusion* (consequent)
  - Conditionals, “if-then” statements, or *rules*
- **Equivalence** $\Leftrightarrow$
  - Binary op., e.g. P $\Leftrightarrow$ Q, “P equivalent to Q”
  - Biconditionals

  "if. else"
- **Negation** $\neg$
  - Unary op., e.g. $\neg$ P, “not P”

Syntax of Propositional Logic
(Backus-Naur Form)

Sentence	$\rightarrow$	AtomicSentence | ComplexSentence
AtomicSentence	$\rightarrow$	LogicalConstant | PropositionalSymbol
ComplexSentence	$\rightarrow$	(Sentence) | Sentence LogicalConnective Sentence | $\neg$Sentence
LogicalConstant	$\rightarrow$	TRUE | FALSE
PropositionalSymbol	$\rightarrow$	P | Q | R | …
LogicalConnective	$\rightarrow$	$\wedge$ | $\vee$ | $\Leftrightarrow$ | $\Rightarrow$ | $\neg$

Precedence (from highest to lowest): $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

e.g.: $\neg P \wedge Q \vee R \Rightarrow S$ (not ambiguous), equal to: $(((\neg P) \wedge Q) \vee R) \Rightarrow S$

Semantics of Propositional Logic

- **Validity**
  - A sentence is valid if it is true in all models.
  - Valid sentences are known as **tautologies**.
  - Every valid sentence is logically equivalent to *True*.

Semantics of Propositional Logic

- **Satisfiability**
  - A sentence is satisfiable if it is true in some models.
  - Satisfiability can be checked by enumerating the possible models until one is found that satisfies the sentence.
  - Most problems in computer sciences are satisfiability problems.
    - E.g., Constraint satisfaction problem, Search problems.

Semantics of Propositional Logic

- **Interpretation of symbols**
  - Logical constants have fixed meaning
    - True: always means the fact is the case; valid
    - False: always means the fact is not the case; unsatisfiable
  - Propositional symbols mean "whatever they mean"
    - e.g.: **P** "we are in a pit", etc.
    - Satisfiable, but not valid (true only when the fact is the case)
- **Interpretation of sentences**
  - Meaning derived from the meaning of its parts
    - Sentence as a combination of sentences using connectives
  - Logical connectives as (boolean) functions:

TruthValue ***f*** (TruthValue, TruthValue)

Example 1

- Let P stands for Intelligent(Anil)
- Let Q stands for Hardworking(Anil)

- What does P Λ Q mean? Anil is intelligent & hardworking
- What does P $\vee$ Q mean? " " or "

- P Λ Q, P $\vee$ Q are compound propositions

Example 2

- Use parenthesis to ensure that the syntax is completely unambiguous:
  - A: John likes Kate.
  - B: John likes Chocolate.
  - C: John buys Chocolate

- $(A \wedge B) \Rightarrow C$ Rule
  - If John likes Kate and John likes Chocolate, John buys Chocolate
- $A \wedge (B \Rightarrow C)$ Two Facts.
  - John likes Kate, and
  - If John likes Chocolate, then John buys Chocolate

Semantics of Propositional Logic

- **Interpretation of connectives**
  - Truth-table
    - Define a mapping from input to output Whenever P is TRUE Q IS TRUE

P	Q	$\neg P$	$P \land Q$	$P \lor Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
False	False	True	False	False	True	True
False	True	True	False	True	True	False
True	False	False	False	True	False	False
True	True	False	True	True	True	True

  - Interpretation of sentences by decomposition
    - e.g.: $\neg P \land Q \lor R \Rightarrow S$, with $P \leftarrow F, Q \leftarrow T, R \leftarrow F, S \leftarrow T$:

$\neg P \leftarrow T$ $\quad ((\neg P) \land Q) \lor R) \leftarrow T$

$(\neg P) \land Q \leftarrow T$ $\quad (((\neg P) \land Q) \lor R) \Rightarrow S \leftarrow T$

Validity and Inference

- **Testing for validity**
  - Using truth-tables, checking all possible configurations
    - e.g.: $((P \lor Q) \land \neg Q) \Rightarrow P$

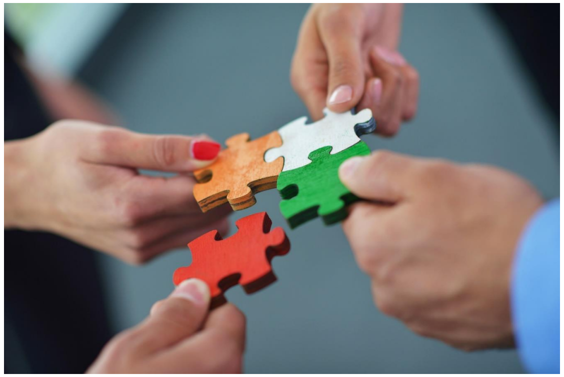
P	Q	$P \lor Q$ (OR)	$\neg Q$	$(P \lor Q) \land \neg Q$	$((P \lor Q) \land \neg Q) \Rightarrow P$
False	False	False	True	False	True
False	True	True	False	False	True
True	False	True	True	True	True
True	True	True	False	False	True

- The proposition says:
  - If $((P \lor Q) \land \neg Q)$ is True, then P is True.
  - If $((P \lor Q) \land \neg Q)$ is False, then ? (didn't specify, so P can be either True or False) -> overall, this proposition is *valid*

Exercise

A	B	C	$A \land B$	$B \Rightarrow C$	$(A \land B) \Rightarrow C$	$A \land (B \Rightarrow C)$
T	T	T				
T	T	F				
T	F	T				
T	F	F				
F	T	T				
F	T	F				
F	F	T				
F	F	F				

Thank you!



NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

CZ3005 Artificial Intelligence

Week 9b – Propositional Logic

Yu Han

han.yu@ntu.edu.sg

Nanyang Assistant Professor
School of Computer Science and Engineering
Nanyang Technological University

Recap

Sentence	→	AtomicSentence | ComplexSentence
AtomicSentence	→	LogicalConstant | PropositionalSymbol
ComplexSentence	→	(Sentence) | Sentence LogicalConnective Sentence | ¬Sentence
LogicalConstant	→	TRUE | FALSE
PropositionalSymbol	→	P | Q | R | …
LogicalConnective	→	$\wedge$ | $\vee$ | $\Leftrightarrow$ | $\Rightarrow$ | $\neg$

Precedence (from highest to lowest): $\neg, \wedge, \vee, \Rightarrow, \Leftrightarrow$

e.g.: $\neg P \wedge Q \vee R \Rightarrow S$ (not ambiguous), equal to: $(((\neg P) \wedge Q) \vee R) \Rightarrow S$

Recap

A	B	C	A ∧ B	B ⇒ C	(A ∧ B) ⇒ C	A ∧ (B ⇒ C)
T	T	T				
T	T	F				
T	F	T				
T	F	F				
F	T	T				
F	T	F				
F	F	T				
F	F	F				

Recap

A	B	C	A ∧ B	B ⇒ C	(A ∧ B) ⇒ C	A ∧ (B ⇒ C)
T	T	T	T	T	T	T
T	T	F	T	F	F	F
T	F	T	F	T	T	T
T	F	F	F	T	T	T
F	T	T	F	T	T	F
F	T	F	F	F	T	F
F	F	T	F	T	T	F
F	F	F	F	T	T	F

Literal and Clause

- **Literal**: A single proposition or its negation:
  - Example: P, ¬P
- A **clause**: A propositional formula formed from a finite collection of literals and logical connectives:
  - Example: $P \vee Q \vee \neg R$

Rules of Inference

- **Sound inference rules**
  - Pattern of inference, that occur again and again
  - Soundness proven once and for all (truth-table)

$$\frac{\alpha}{\beta}$$

- **Classic rules of inference**
  - Implication-Elimination, or *Modus Ponens (MP)*
    - $$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

e.g., Cloudy Λ Humid $\Rightarrow$ Rain
Cloudy Λ Humid |= Rain

Rules of Inference

- **Classic rules of inference**
  - And-Elimination
    - $$\frac{\alpha_1 \Lambda \alpha_2 \Lambda \ldots \Lambda \alpha_n}{\alpha_i}$$

    e.g. Cloudy Λ Humid |= Cloudy
    Cloudy $\Rightarrow$ NoSunTan
  - And-Introduction
    - $$\frac{\alpha_1, \alpha_2, \ldots, \alpha_n}{\alpha_1 \Lambda \alpha_2 \Lambda \ldots \Lambda \alpha_n}$$

    e.g. Cloudy, Humid
    Cloudy Λ Humid $\Rightarrow$ Rain
  - Or-Introduction
    - $$\frac{\alpha_i}{\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_n}$$
  - Double-Negation-Elimination
    - $$\frac{\neg\neg\alpha}{\alpha}$$

Rules of Inference

- **Resolution**
  - A technique of inference
  - Suppose *x* is a literal and S1 and S2 are two propositional sentences represented in the clausal form
- If $(x \vee S1)\ \Lambda\ (\neg x \vee S2)$. Then, we get $(S1 \vee S2)$
  - Here, $(S1 \vee S2)$ is the resolvent,
  - x is resolved upon

Rules of Inference

- **The resolution rule of inference**
  - Unit Resolution
  - Unit resolution is a specific case of resolution where one of the clauses involved has only a single literal.

$$\frac{\alpha \vee \beta,\ \neg\beta}{\alpha}$$

e.g., Monday $\vee$ Tuesday, $\neg$ Monday |= Tuesday

same as MP: $\dfrac{P \Rightarrow Q,\ P}{Q}$ i.e. $\dfrac{\neg\beta \Rightarrow \alpha,\ \neg\beta}{\alpha}$

Rules of Inference

- **The resolution rule of inference**
  - Full Resolution (a more general form of the resolution rule)
    - $\dfrac{\alpha \vee \beta,\ \neg\beta \vee \gamma}{\alpha \vee \gamma}$

Truth-table for the resolution

α	β	γ	$\alpha \vee \beta$	$\neg\beta \vee \gamma$	$\alpha \vee \gamma$
False	False	False	False	True	False
False	False	True	False	True	True
False	True	False	True	False	False
False	True	True	True	True	True
True	False	False	True	True	True
True	False	True	True	True	True
True	True	False	True	False	True
True	True	True	True	True	True

Equivalence Rules

- **Equivalent notations**
  - e.g., MP:

$$\frac{\alpha \Rightarrow \beta,\ \alpha}{\beta}$$

1) $\alpha \Rightarrow \beta, \alpha \ \vdash \ \beta$
2) $\alpha \Rightarrow \beta, \alpha \ \models \ \beta$
3) $\dfrac{\begin{array}{c}\alpha \Rightarrow \beta \\ \alpha\end{array}}{\beta}$
4) $((\alpha \Rightarrow \beta) \wedge \alpha) \Rightarrow \beta$

Equivalence Rules

- **Equivalence rules**
  - Associativity:
    $\alpha \wedge (\beta \wedge \gamma) \Leftrightarrow (\alpha \wedge \beta) \wedge \gamma$
    $\alpha \vee (\beta \vee \gamma) \Leftrightarrow (\alpha \vee \beta) \vee \gamma$
  - Distributivity:
    $\alpha \wedge (\beta \vee \gamma) \Leftrightarrow (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
    $\alpha \vee (\beta \wedge \gamma) \Leftrightarrow (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$
  - De Morgan's Law:
    $\neg(\alpha \vee \beta) \Leftrightarrow \neg\alpha \wedge \neg\beta$
    $\neg(\alpha \wedge \beta) \Leftrightarrow \neg\alpha \vee \neg\beta$

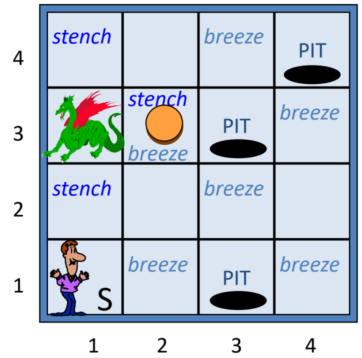
Complexity of Inference

- **Proof by truth-table**
  - Complete
    - The truth-table can always be written.
  - Exponential time complexity
    - A proof involving ***N*** proposition symbols requires $\mathbf{2^N}$ rows.
    - In practice, a proof may refer only to a small subset of the KB.
- **Monotonicity**
  - Knowledge always increases

$$\textbf{if}\ \ KB_1 \models \alpha\ \ \textbf{then}\ \ (KB_1 \cup KB_2) \models \alpha$$

Relooking at the Wumpus World

- **A reasoning agent**
  - Propositional logic as the "programming language"
  - Knowledge base (KB) as problem representation
    - Percepts
    - Knowledge — sentences
    - Actions
  - Rule of inference (e.g., Modus Ponens) as the algorithm that will find a solution



The Knowledge Base

- **TELLing the KB: percepts**
  - Syntax and semantics
    - Symbol **S11**, meaning "there is stench at [1,1]"
    - Symbol **B12**, meaning "there is breeze at [1,2]"
    …
- **Percept sentences**
  - Partial list:
    - ¬ S11, ¬ B11, ¬ G11, …
      ¬ S21, B21, ¬ G21, …
      S12, ¬ B12, ¬ G12, …
      …

[Stench , nil, nil, nil, nil]

	1	2	3	4
4				
3	W!			
2	A S OK	OK		
1	V OK	B V OK	P!	

The Knowledge Base

- **TELLing the KB: knowledge**
  - Rules about the environment
    - "All squares adjacent to the wumpus have stench."
      $S12 \Rightarrow W11 \lor W12 \lor W22 \lor W13$
    - "A square with no stench has no wumpus and adjacent squares have no wumpus either."
      $\neg S11 \Rightarrow \neg W11 \land \neg W21 \land \neg W12$
      $\neg S21 \Rightarrow \neg W11 \land \neg W21 \land \neg W22 \land \neg W31$
      $\neg S12 \Rightarrow \neg W11 \land \neg W12 \land \neg W22 \land \neg W13$

[Stench , nil, nil, nil, nil]

	1	2	3	4
4				
3	W!			
2	A S OK	OK		
1	V OK	B V OK	P!	

Finding the Wumpus

- **Checking the truth-table**
  - Exhaustive check: every row for which KB is true also has W13 true
    - 12 propositional symbols, i.e. S11, S21, S12, W11, W21, W12, W22, W13, W31, B11, B21, B12
    - 2^{12} = 4,096 rows
  - > *possible, but lengthy*

KB $\Rightarrow$ W13

- **Reasoning by inference**
  - Application of a sequence of inference rules (proof)
    - Modus Ponens, And-Elimination, and Unit-Resolution

Proof for "KB $\Rightarrow$ W13"

Knowledge Base	Inferences
$\neg$ S11, $\neg$ B11, $\neg$ G11, $\neg$ S21, B21, $\neg$ G21, S12, $\neg$ B12, $\neg$ G12, **R1:** $\neg S11 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W12$ **R2:** $\neg S21 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$ **R3:** $\neg S12 \Rightarrow \neg W11 \wedge \neg W12 \wedge \neg W22 \wedge \neg W13$ **R4:** $S12 \Rightarrow W11 \vee W12 \vee W22 \vee W13$	1. Modus Ponens: $\neg$ S11, **R1** $\vdash$ $\neg W11 \wedge \neg W21 \wedge \neg W12$ 2. And-Elimination: ♦ $\vdash$ $\neg W11$, $\neg W21$, $\neg W12$ 3. Modus Ponens: $\neg$ S21, **R2** $\vdash$ $\neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$ 4. And-Elimination: ♦ $\vdash$ $\neg W11$, $\neg W21$, $\neg W22$, $\neg W31$

Proof for "KB $\Rightarrow$ W13"

Knowledge Base	Inferences
$\neg$ S11, $\neg$ B11, $\neg$ G11, $\neg$ S21, B21, $\neg$ G21, S12, $\neg$ B12, $\neg$ G11, **R1:** $\neg S11 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W12$ **R2:** $\neg S21 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$ **R3:** $\neg S12 \Rightarrow \neg W11 \wedge \neg W12 \wedge \neg W22 \wedge \neg W13$ **R4:** $S12 \Rightarrow W11 \vee W12 \vee W22 \vee W13$	KB += $\neg W11$, $\neg W21$, $\neg W12$, $\neg W22$, $\neg W31$ 5. Modus Ponens: S12, **R4** $\vdash$ $(W13 \vee W12 \vee W22) \vee W11$ 6. Unit-Resolution: ♦, $\neg W11$ $\vdash$ $(W13 \vee W12) \vee W22$

Proof for "KB $\Rightarrow$ W13"

Knowledge Base	Inferences
$\neg$ S11, $\neg$ B11, $\neg$ G11, $\neg$ S21, B21, $\neg$ G21, S12, $\neg$ B12, $\neg$ G12, **R1:** $\neg S11 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W12$ **R2:** $\neg S21 \Rightarrow \neg W11 \wedge \neg W21 \wedge \neg W22 \wedge \neg W31$ **R3:** $\neg S12 \Rightarrow \neg W11 \wedge \neg W12 \wedge \neg W22 \wedge \neg W13$ **R4:** $S12 \Rightarrow W11 \vee W12 \vee W22 \vee W13$	KB += $\neg W11$, $\neg W21$, $\neg W12$, $\neg W22$, $\neg W31$, $(W13 \vee W12) \vee W22$ 7. Unit-Resolution: ♦, $\neg W22$ $\vdash$ $W13 \vee W12$ 8. Unit-Resolution: ♦, $\neg W12$ $\vdash$ W13 **KB $\Rightarrow$ W13**

From Knowledge to Actions

- **TELLing the KB: actions**
  - Additional rules
    - e.g. "if the wumpus is 1 square ahead then do not go forward"

      A12 ∧ East ∧ W22 ⇒ ¬Forward

      A12 ∧ North ∧ W13 ⇒ ¬Forward

      …
- **ASKing the KB**
  - Cannot ask "which action?" but "should I go forward?"

[Stench , nil, nil, nil, nil]

4				
3	W!			
2	A S OK	OK		
1	V OK	B V OK	P!	
	1	2	3	4

A Knowledge-Based Agent Using Propositional Logic

```
function Propositional-KB-Agent (percept) returns action
        static   KB,            // a knowledge base
                 t              // a time counter, initially 0

        Tell (KB, Make-Percept-Sentence (percept, t))
        foreach action in the list of possible actions
        do
           if Ask (KB, Make-Action-Query (t, action)) then
                 Tell (KB, Make-Action-Sentence (action, t))
                 t ← t + 1
                 return action
        end
```


Limits of Propositional Logic

- **A weak logic**
  - Too many propositions to TELL the KB
    - e.g., the rule "if the wumpus is 1 square ahead then do not go forward" needs 64 sentences (16 squares x 4 orientations)!
    - Result in increased time complexity of inference
  - Handling change is difficult
    - Need time-dependent propositional symbols
      e.g., A11 means "the agent is in square [1,1]" - when?
      at t = 0: A11-0; at t = 1: A21-1;
      at t = 2: A11-2
    - Need to rewrite rules as time-dependent
      e.g., A12-0 ∧ East-0 ∧ W22-0 ⇒ ¬Forward-0
      A12-2 ∧ East-2 ∧ W22-2 ⇒ ¬Forward-2

Thank you!