Logic

- Use mathematical deduction to derive new knowledge.
- Predicate Logic is a powerful representation scheme used by many AI programs.
- Propositional logic is much simpler (less powerful).

Propositional Logic

- Symbols represent *propositions* (facts).
- A proposition is either *TRUE* or *FALSE*.
- Boolean connectives can join propositions together into complex *sentences*.
- Sentences are statements that are either *TRUE* or *FALSE*.

Propositional Logic Syntax

- The constants *TRUE* and *FALSE*.
- Symbols such as *P* or *Q* that represent propositions.
- Logical connectives:

Ù AND, conjunction

Ú OR, disjunction

Þ Implication , conditional (If then)

Û Equivalence , biconditional

Ø Negation (unary)

( ) parentheses (grouping)

Truth Tables

P	Q	Ø P	PÙQ	PÚQ	PÞ Q	PÛ 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

Sentences

- *True*, *False* or any proposition symbol is a sentence.
- Any sentence surrounded by parentheses is a sentence.
- The disjunction, conjunction, implication or equivalence of 2 sentences is a sentence.
- The negation of a sentence is a sentence.

Examples

(P Ú Q) Þ R — If P or Q is true, then R is true

P Û (Q Ù R) — If Q and R are both true, P must be true AND if Q or R is false then P must be false.

Ø P Þ (Q Þ R) — If P is false, then If Q is true R must be true.

Sentence Validity

- A propositional sentence is valid (TRUE) if and only if it is true under all possible interpretations in all possible domains.
- For example:

  `If Today_Is_Tuesday Then We_Have_Class`

  The truth does not depend on whether today is Tuesday but on whether the relationship is true.

Inference Rules

- There are many patterns that can be formally called *rules of inference* for propositional logic.
- These patterns describe how new knowledge can be derived from existing knowledge, both in the form of propositional logic sentences.
- Some patterns are common and have fancy names.

Inference Rule Notation

- When describing an inference rule, the *premise* specifies the pattern that must match our knowledge base and the *conclusion* is the new knowledge inferred.
- We will use the notation

$$\textit{premise} \;\vdash\; \textit{conclusion}$$

Inference Rules

- Modus Ponens: $x \Rightarrow y, x \vdash y$
- And-Elimination: $x_1 \wedge x_2 \wedge \ldots \wedge x_n \vdash x_i$
- And-Introduction: $x_1, x_2, \ldots, x_n \vdash x_1 \wedge x_2 \wedge \ldots \wedge x_n$
- Or-Introduction: $x \vdash x \vee y \vee z \vee \ldots$
- Double-Negation Elimination: $\neg\neg x \vdash x$
- Unit Resolution: $x \vee y, \neg x \vdash y$

Resolution Inference Rule

$$x \vee y,\ \neg y \vee z \ \vdash\ x \vee z$$

-or-

$$\neg x \Rightarrow y,\ y \Rightarrow z \ \vdash\ \neg x \Rightarrow z$$

Logic & Finding a Proof

- Given
  - a knowledge base represented as a set of propositional sentences.
  - a goal stated as a propositional sentence
  - a list of inference rules
- We can write a program to repeatedly apply inference rules to the knowledge base in the hope of deriving the goal.

Example

It will snow OR there will be a test.

Dave is Darth Vader OR it will not snow.

Dave is not Darth Vader.

Will there be a test?

Solution

Snow = *a* Test = *b* Dave is Vader = *c*

Knowledge Base (these are all true):

a **Ú** *b*, *c* **Ú** Ø *a*, Ø c

By Resolution we know *b* **Ú** *c* is true.

By Unit Resolution we know *b* is true.

There will be a test!

Developing a Proof Procedure

- Deriving (or refuting) a goal from a collection of logic facts corresponds to a very large search tree.
- A large number of *rules of inference* could be utilized.
- The selection of which rules to apply and when would itself be non-trivial.

Resolution & CNF

- *Resolution* is a single rule of inference that can operate efficiently on a special form of sentences.
- The special form is called *clause form* or *conjunctive normal form* (CNF), and has these properties:
  – Every sentence is a disjunction (or) of literals
  – All sentences are implicitly conjuncted (anded).

Propositional Logic and CNF

- Any propositional logic sentence can be converted to CNF. We need to remove all connectives other than OR (without modifying the meaning of a sentence)

Converting to CNF

- Eliminate implications and equivalence.
- Reduce scope of all negations to single term.
- Use associative and distributive laws to convert to a conjunct of disjuncts.
- Create a separate sentence for each conjunct.

Eliminate Implications and Equivalence

$$x \Rightarrow y \quad \text{becomes} \quad \neg x \vee y$$

$$x \Leftrightarrow y \quad \text{becomes} \quad (\neg x \vee y) \wedge (\neg y \vee x)$$

Reduce Scope of Negations

Ø (Ø x) becomes x

Ø (x **Ú** y) becomes (Ø y **Ù** Ø x)

Ø (x **Ù** y) becomes (Ø y **Ú** Ø x)

deMorgans Laws

Convert to conjunct of disjuncts

Associative property:

(A v B) v C = A v (B v C)

Distributive property:

(A ^ B) v C = (A v C) ^ (B v C)

Using Resolution to Prove

- Convert all propositional sentences that are in the knowledge base to CNF.
- Add the <u>contradiction</u> of the goal to the knowledge base (in CNF).
- Use resolution as a rule of inference to prove that the combined facts can not all be true.

Proof by contradiction

- We assume that all original facts are TRUE.
- We add a new fact (the contradiction of sentence we are trying to prove is TRUE).
- If we can infer that FALSE is TRUE we know the knowledgebase is corrupt.
- The only thing that might not be TRUE is the negation of the goal that we added, so if must be FALSE. Therefore the goal is true.

Propositional Example: The Mechanics of Proof

Knowledge Base:

P

(*P* Ù *Q*) Þ R

(S Ú T) Þ Q

T

These represent the facts we know to be true.

Goal:

R

This is what we want to prove is true.

Conversion to CNF

<u>Sentence</u>	<u>CNF</u>
P	**P**
(*P* Ù *Q*) Þ R	**Ø P Ú Ø Q Ú R**
(S Ú T) Þ Q	**Ø S Ú Q**
	Ø T Ú Q
T	**T**

Add Contradiction of Goal

- The goal is **R**, so we add Ø **R** to the list of facts, the new set is:

  1. **P**
  2. Ø **P Ú** Ø **Q Ú R**
  3. Ø **S Ú Q**
  4. Ø **T Ú Q**
  5. **T**
  6. Ø **R**

Resolution Rule of Inference

Recall the general form of resolution:

$$x_1 \vee x_2 \vee ... \vee x_n \vee z, \quad y_1 \vee y_2 \vee \ldots y_m \vee \neg z \vdash$$
$$x_1 \vee x_2 \vee ... \vee x_n \vee y_1 \vee y_2 \vee \ldots y_m$$

Applying Resolution

Fact 2 can be *resolved* with fact 6, yielding a new fact:

Ø P Ú Ø Q Ú R **Ø R**

Ø P Ú Ø Q

A new fact, call it fact 7.

2. ∅ **P Ú ∅ Q Ú R**   6. ∅ **R**

7. ∅ **P Ú ∅ Q**   1. **P**

4. ∅ **T Ú Q**   8. ∅ **Q**

9. ∅ **T**   5. **T**

ð

Null Clause

There is no way all the clauses can be true!

A more intuitive look at the same example.

P: Joe is smart

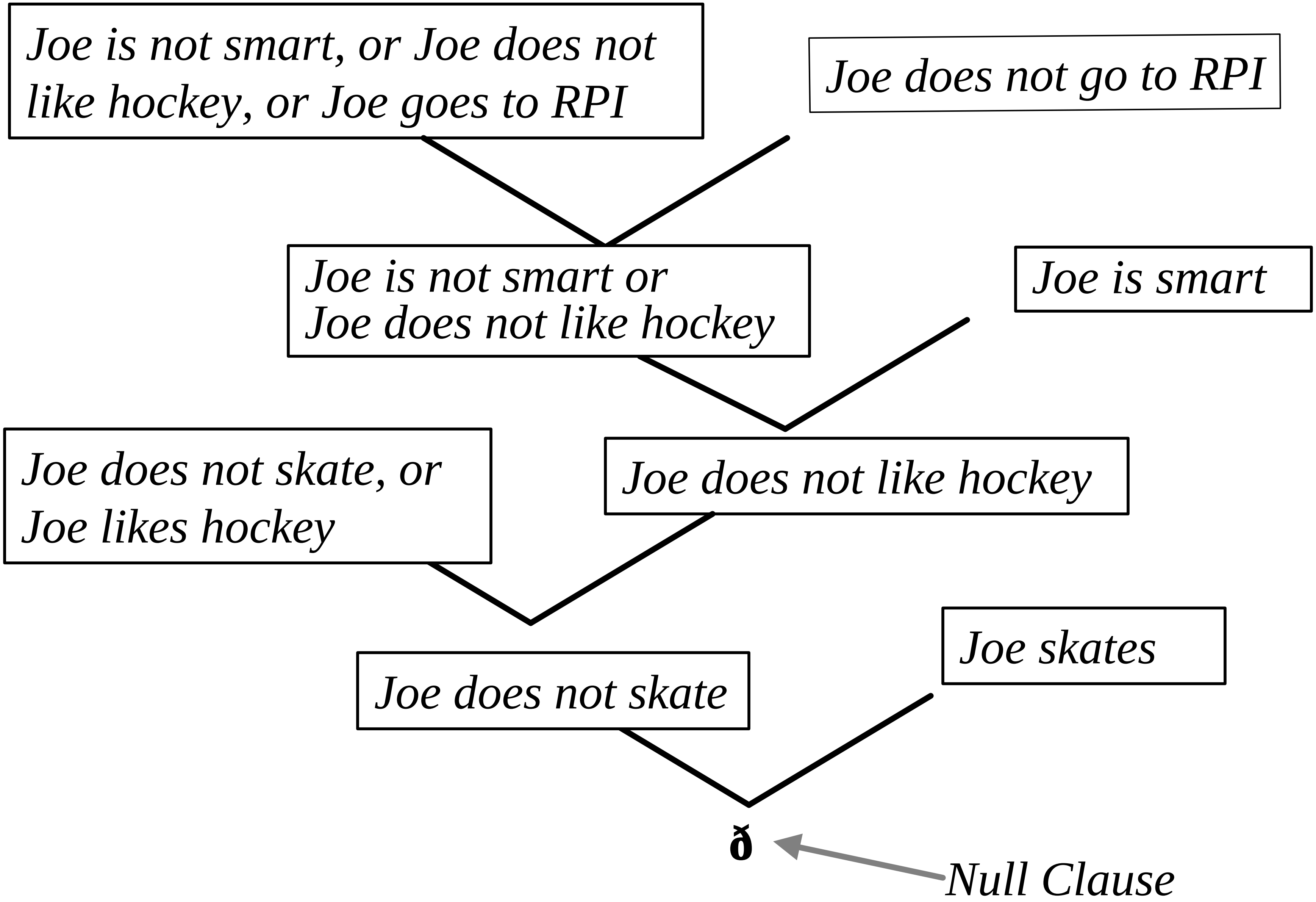
Q: Joe likes hockey

R: Joe goes to RPI

S: Joe is Canadian

T: Joe skates.

- Original Sentences:
  - Joe is smart
  - If Joe is smart and Joe likes hockey, Joe goes to RPI
  - If Joe is Canadian or Joe skates, Joe likes hockey.
  - Joe skates.
- After conversion to CNF:
  - Fact 2: Joe is not smart, or Joe does not like hockey, or Joe goes to RPI.
  - Fact 3: Joe is not Canadian or Joe likes hockey.
  - Fact 4: Joe does not skate, or Joe likes hockey.


Joe is not smart, or Joe does not like hockey, or Joe goes to RPI
Joe does not go to RPI
Joe is not smart or Joe does not like hockey
Joe is smart
Joe does not skate, or Joe likes hockey
Joe does not like hockey
Joe skates
Joe does not skate
ð
Null Clause

Propositional Logic Limits

- The expressive power of propositional logic is limited. The assumption is that everything can be expressed by simple facts.
- It is much easier to model real world objects using *properties* and *relations*.
- Predicate Logic provides these capabilites more formally and is used in many AI domains to represent knowledge.

Propositional Logic Problem

If the unicorn is mythical, then it is immortal, but if it is not mythical then it is a mortal mammal. If the unicorn is either immortal or a mammal, then it is horned. The unicorn is magical if it is horned.

Q: Is the unicorn mythical? Magical? Horned?