

# Knowledge Representation and Reasoning

Propositional Calculus  
and Resolution

# Propositional Calculus

- What is Propositional Calculus (P.C.)?
- Elements of P.C. – Formal Language
- Reasoning: Proofs and Entailments
- Clauses and Resolution
- Resolution as Inference Rule
- Algorithms and Complexity

# Formal P.C. Language

Elements of P.C. language:

- Atoms: P, R, Q, A<sub>2</sub>, ...
- Connectives:  $\wedge$  ,  $\vee$  ,  $\supset$  ,  $\neg$  (AND, OR, IMPLIES, NOT)
- Well-formed formulas (wff)  
 $P$  ,  $R \wedge A_1$  ,  $P_1 \supset (\neg B)$  , ...

# Formal P.C. Language

The Propositional Truth Table:

P1	P2	$P1 \wedge P2$	$P1 \vee P2$	$\neg P1$	$P1 \supset P2$
True	True	True	True	False	True
True	False	False	True	False	False
False	True	False	True	True	True
False	False	False	False	True	True

Note: Semantics of  $(P1 \supset P2)$  is equivalent to  $(\neg P1 \vee P2)$

# Formal P.C. Language

Rules of Inference:

- $\{P_1, P_2\} \rightarrow P_1 \wedge P_2$
- $\{P_1\} \rightarrow P_1 \vee (\text{any})$
- $\{P_1, (P_1 \supset P_2)\} \rightarrow P_2$       “*modus ponens*”
- $\{\neg \neg P_1\} \rightarrow P_1$
- .....

# Formal P.C. Language

## Proofs:

D = sequence of wff (prior knowledge)

- E = “theorem”, a wff to be proven
- Proof:  $D \rightarrow E$  (deduction rule)
- Example:  $D=\{P, R, P \supset Q\}$  ,  $E=\{Q \wedge R\}$

$$D=\{P, R, P \supset Q\} \rightarrow \{P, P \supset Q, Q, R, Q \wedge R\} \rightarrow \{Q \wedge R\} = E$$

## Entailment:

- “Discover” wff that hold true given D

# Formal P.C. Language

Equivalence of wff:

- Produce the same outcome in all cases
- Formal definition:

$$P_1 \equiv P_2 : (P_1 \supset P_2) \wedge (P_2 \supset P_1)$$

# Formal P.C. Language

## Soundness:

- If any  $P_1$  that is implied by  $D$  and rules  $R$  can be “discovered” by entailment

## Completeness:

- If any  $P_1$  that is implied by  $D$  and rules  $R$  can be “proven” by a proof

- The PSAT (Propositional Satisfiability) Problem: find a model  $D$  that implies a given formula  $P_1$
- Common situation in circuit design, path planning, etc.
- Usual form: CNF – Conjunctive Normal Form

# Resolution in P.C.

- Resolution: use deduction rules to assert or discard the validity of a Clause.
- Clause: any formatted wff that is used in a Resolution scheme.
- Resolution on Clauses:
  - Follow 3 simple rules for converting any wff into a clause in CNF that can be resolved

# Resolution in P.C.

- Converting a wff to a clause (CNF):

$$\neg(P \supset Q) \vee (R \supset P)$$

1. Eliminate implication signs ( $\supset$ ) by using the equivalent form using ( $\neg\vee$ ):

$$\neg(\neg P \vee Q) \vee (\neg R \vee P)$$

2. Reduce the ( $\neg$ ) signs using De Morgan's laws:

$$(P \wedge \neg Q) \vee (\neg R \vee P)$$

3. Convert to CNF, i.e. place ( $\wedge$ ) outside parentheses:

$$(P \vee \neg R \vee P) \wedge (\neg Q \vee \neg R \vee P) \equiv \{(P \vee \neg R), (\neg Q \vee \neg R \vee P)\}$$

# Resolution in P.C.

## Resolution as Inference Rule:

- Resolution is “sound” BUT not “complete”, i.e. not all logical expressions can be entailed.
- Instead, formulate the negation of the clause to be entailed and then investigate if this new clause can be proven within the current “state of world”.
- Resolution Refutation: proof of the negation (non-empty result) invalidates the original clause, otherwise the original clause is asserted as true.

# P.C. Resolution Strategies

Problem: In what order should the resolutions be performed for optimal results?

- Breadth-first: expand all nodes in same level
- Depth-first: expand each node to the end
- Unit-preference: expand “small” nodes first
- Horn clauses: contain at most one positive literal  
⇒ limits the complexity of deduction search to linear times.

## P.C. – Readings

- Nils J. Nilsson, “Artificial Intelligence – A New Synthesis”, Morgan Kaufmann Publishers (1998).  
[see: ch.13 & ch.14]
- S. J. Russell, P. Norvig, “Artificial Intelligence: A Modern Approach”, 2nd/Ed, Prentice Hall, 2002.