CS 344

Artificial Intelligence

By Prof: Pushpak Bhattacharya

Class on 15/Feb/2007

Completeness of Propositional Calculus

- Statement

  If $V(A) = T$ for all V,

  then $|\text{--}A$ *i.e.* A is a theorem.

- Lemma:

  If A consists of propositions $P_1, P_2, \ldots, P_n$ then $P'_1, P'_2, \ldots, P'_n \;|\text{--}\; A'$, where

  $A' = A$ if $V(A) = true$

  $= \sim A$ otherwise

  Similarly for each P'_i

Proof for Lemma

- Proof by induction on the number of '$\rightarrow$' symbols in *A*

  Basis: Number of '$\rightarrow$' symbols is zero.

  A is $\mathscr{F}$ or *P*. This is true as, |-- $(A \rightarrow A)$

  i.e. $A \rightarrow A$ is a theorem.

  Hypothesis: Let the lemma be true for number of '$\rightarrow$' symbols $\leq n$.

  Induction: Let *A* which is $B \rightarrow C$ contain $n+1$ '$\rightarrow$'

Proof of Lemma (contd.)

- Induction:
  By hypothesis,
  $P'_1, P'_2, \ldots, P'_n$ |-- B'
  $P'_1, P'_2, \ldots, P'_n$ |-- C'
  If we show that B', C' |-- A' *(A is* $B \rightarrow C$*)*, then the proof is complete.
  For this we have to show:
  - B, C |-- $B \rightarrow C$
    True as B, C, B |-- C
  - $B, \sim C$ |-- $B \rightarrow C$
    True since $B, \sim C, B \rightarrow C$ |-- $\mathcal{F}$
  - $\sim B, C$ |-- $B \rightarrow C$
    True since $\sim B, C, B$ |-- C
  - $\sim B, \sim C$ |-- $B \rightarrow C$
    True since $\sim B, \sim C, B, C, C \rightarrow \mathcal{F}$ |-- $\mathcal{F}$
- Hence the lemma is proved.

Proof of Theorem

- A is a tautology.
- There are 2^n models corresponding to $P_1, P_2, \ldots, P_n$ propositions.
- Consider,

$$P_1, P_2, \ldots, P_n \;\; |\text{--} \;\; A$$

and

$$P_1, P_2, \ldots, \sim P_n \;\; |\text{--} \;\; A$$

$$P_1, P_2, \ldots, P_{n-1} \;\; |\text{--} \;\; P_n \rightarrow A$$

and

$$P_1, P_2, \ldots, P_{n-1} \;\; |\text{--} \;\; \sim P_n \rightarrow A$$

RHS can be written as:

$$|\text{--} \;\; ((P_n \rightarrow A) \rightarrow ((\sim P_n \rightarrow A) \rightarrow A))$$

$$|\text{--} \;\; (\sim P_n \rightarrow A) \rightarrow A$$

$$|\text{--} \;\; A$$

- Thus dropping the propositions progressively we show |-- A

Detour

- Reasoning
  - two types:
    - Monotonic: Adding inferred knowledge monotonically to the system but not retracting from the knowledge base.
    - Nonmonotonic: Retracts knowledge which becomes false in the face of new evidence
- Types of Sentences in English: 3 kinds of sentences important from Natural Language Processing point of view. Useful to remember in *knowledge extraction*.
  - Simple Sentence: Single verb, *e.g., Ram plays cricket*
  - Compound Sentence: Two independent clauses joined by coordinator. Two verbs are present.

    e.g. Ram went to school and Shyam played cricket
  - Complex Sentence: Independent clauses joined by one or more dependent clauses. More than one verb

    e.g. Ram who sings well is performing in the festival today

Predicate Calculus

- Introduction through an example *(Zohar Manna, 1974):*
  - Problem: A, B and C belong to the Himalayan club. Every member in the club is either a mountain climber or a skier or both. A likes whatever B dislikes and dislikes whatever B likes. A likes rain and snow. No mountain climber likes rain. Every skier likes snow. *Is there a member who is a mountain climber and not a skier?*
- Given knowledge has:
  - Facts
  - Rules

Predicate Calculus: Example contd.

- Let *mc* denote mountain climber and *sk* denotes skier. Knowledge representation in the given problem is as follows:
  1. *member(A)*
  2. *member(B)*
  3. *member(C)*
  4. *□x[member(x) → (mc(x) □ sk(x))]*
  5. *□x[mc(x) → ~like(x,rain)]*
  6. *□x[sk(x) → like(x, snow)]*
  7. *□x[like(B, x) → ~like(A, x)]*
  8. *□x[~like(B, x) → like(A, x)]*
  9. *like(A, rain)*
  10. *like(A, snow)*
  11. *Question: □x[member(x) □ mc(x) □ ~sk(x)]*
- We have to infer the 11^{th} expression from the given 10.
- Done through Resolution Refutation.