Logical Representation

A candidate for the standard "language of thought"

Predicate Logic & First Order Predicate Calculus

- A language for expressing facts and rules,
  - and
  -
- A mathematics for manipulating truth values and well-formed expressions
Uses of Logic in AI

- Representation
  - of problem states
  - of rules
- Reasoning
  - Sound Problem Solving
  - Theorem proving systems
  - Refutation
- Deductive Data Retrieval
- Expert Systems
- Inference

Three Brands of Inference

Abduction
Induction
Deduction
**Abduction**
- Approximate Reasoning
- Plausible Inference
- Reverse Causal Reasoning
  - if P -> Q, and Q, Abduce P
- Not logically sound, but economical

**Induction**
(Physicist's Favorite)
- To Reason from specific cases to general rules
- Very Difficult to Mechanize
- Major problem for Machine Learning
Deduction (as per Sherlock)

- Given a set of
  - Assumptions (Facts)
  - Logical Rules
  - Universal Laws of Logic
- We can find all new facts which logically follow from our assumptions

BACKGROUND

- Logical Expressions are built out of primitives:
  - constants
  - variables
  - predicates
  - connectives
  - quantifiers
- According to well-formedness rules
Propositional vs Predicate Logic

- Propositional Logic works with constants
  - P is True
  - Q is True
  - P \land Q is true

- Predicates are a little more complicated...

Constants

- Symbols which denote objects and individuals in the world:
  - JOHN
  - MARY
  - TABLE
  - BLOCK
**Variables**

- Indefinite References to objects and individuals in the world:
  - x
  - y
  - z

**Predicates (or functions)**

- Functions denoting aspects of objects or relationships between individuals:
  - LOVES(John, Mary)
  - RED(Ball1)
  - INSIDE(x,Box23)
**Connectives**

- Logical operators which compute (or constrain) truth values:
  - & AND
  - | OR
  - ~ NOT
  - => IMPLIES

**Quantifiers**

- Logical Meta-operators which assert across the entire population of objects:
  - ∀ For All
  - ∃ There Exists
**Literals**

- Well-formed expressions which state facts about individuals in the world:
  - Loves(Mary John) & Married(John)
  - Male(Terry) & ~Married(Terry)

- No Variables
- No Quantifiers

**Formulae**

- Well-formed expressions which assert rules (generalized) about relations between objects:
- Sometimes the quantifiers are implicit
  - \( \exists x \) Person(x)
  - \( \forall y \) boy(y) => male (y)
  - \( \forall x \) \( \exists y \) Man(x) & Woman(y) & Loves(x,y)
How to Use Logic?

- Decide on Objects, Predicates
- Translate Facts & Rules about the domain to Expressions
- The Rest is Mechanical...
  - Can represent logical formula in list
  - Can manipulate lists mechanically in LISP to follow logical rules...

AI Extremist Position

- all human thought can be reduced to logic
- All logic can be mechanized
- THEREFORE: all human thought can be mechanized through logic
Problems remain

- Selection (often depends on researcher):
  - Closed World Assumption, Mapping problems
- Translation:
  - Ambiguity, Quantifier Scoping
- Mechanics:
  - Combinatorial Explosion

Mechanics of Logic: Manipulating Expressions without Lying

- Double Negation
  - \( \sim\sim\text{Man}(x) = \text{Man}(x) \)
- DeMorgan's Laws
  - \( \sim(P \& Q) = (\sim P \mid \sim Q) \)
  - \( \sim(P \mid Q) = (\sim P \& \sim Q) \)
Mechanics of Logic:

- **Commutativity**
  - \( P \land Q = Q \land P \)
  - \( P \lor Q = Q \lor P \)

- **Distributivity**
  - \( X \land (Y \lor Z) = (X \land Y) \lor (X \land Z) \)
  - \( X \lor (Y \land Z) = (X \lor Y) \land (X \lor Z) \)

- **Associativity**
  - \( (A \land B) \land C = A \land (B \land C) \)

Mechanics of Logic: Formulae can also be mucked with

- **Demorgans Laws for Quantification**
  - \( \neg \exists x \ P(x) = \forall x \ \neg P(x) \)
  - \( \neg \forall x \ Q(x) = \exists x \ \neg Q(x) \)

- **Skolemization**:
  - Replacing existentials with variable-binding functions
  - Renaming Variables
### Laws of Inference

**Domain-Independent Universally Sound Rules**

- **Modus Ponens (Forward)**
  - If P => Q and P is true, then infer Q
- **Modus Tolens**
  - If P => Q and Q is False, then Infer ~P
- **Syllogism**
  - If P => Q and Q => R then Infer P => R

### Logic as "problem solving"

<table>
<thead>
<tr>
<th>Initial State</th>
<th>Operators</th>
<th>Goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statements of fact, Object</td>
<td>Formulae for domain</td>
<td>Proof of some truth, refutation</td>
</tr>
<tr>
<td>Database</td>
<td>Rules of Inference</td>
<td></td>
</tr>
</tbody>
</table>
Combinatorial Explosion!

- There Are Too Many True Facts
- There Are Too Many Equiv. Representations
- There are Too Many Operators
- It's easy to make infinite loops creating more and more "knowledge spam".
  - $\forall x \exists y \text{person}(x) \Rightarrow \text{person}(y) \& \text{parent}(x,y)$

Resolution

The great hope: One law of inference to mechanize logic
The Resolution Principle

- One way of representing things
  - Conjunctive Normal Form
- One Inference Rule
  - Disjunctive Syllogism

Disjunctive Syllogism
Subsumes other Laws of inference

- Two Disjuncts yield something new
  - \((A \mid B \mid C)\) and \((D \mid \neg B \mid E)\) yields \((A \mid C \mid D \mid E)\)
- Modus Ponens
  - \((P)\) and \((\neg P \mid Q)\) yields \((Q)\)
- Modus Tolens
  - \((\neg Q)\) and \((\neg P \mid Q)\) Yields \((\neg P)\)
- Chaining
  - \((\neg P \mid Q)\) and \((\neg Q \mid R)\) yields \((\neg P \mid R)\)
Conjunctive Normal Form

- Product of Sums
  - A set of Clauses, Conjoined
  - Each Clause is a disjunction

How Do We Get to CNF?
(buy software to do it)

- Eliminate Implication
  - \( A \implies B = (\neg A \lor B) \)
- Reduce Scope of Negation
- Standardize Variables
- Skolemize
- Prenexize
- Remove Universal Quants
- Standardize Variables Apart
Resolution Algorithm

- Until NIL is found
  - Select Two Clauses
  - Compute Their Resolvent
  - Add it to the Set

Proof by Refutation:
- Add the negation of the goal to the set of clauses
- See if you can find a contradiction
- If so, it is proved.
  - Can also be used for deductive data retrieval.
  - Can also be basis for programming language (PROLOG)

How is it used?
Example

- Whoever can read is literate
- Dolphins are not literate
- Some Dolphins are Intelligent
- PROVE:
  - Some who are intelligent can't read

Convert to Predicate Form

- Whoever can read is literate
  - $\forall x \ R(x) \Rightarrow L(x)$
- Dolphins are not literate
  - $\forall x \ D(x) \Rightarrow \neg L(x)$
- Some Dolphins are Intelligent
  - $\exists x \ D(x) \& I(x)$
- PROVE:
  - Some who are intelligent can't read
  - $\exists x \ I(x) \& \neg R(x)$
Ignore Reality

- \( \forall x \ R(x) \Rightarrow L(x) \)
- \( \forall x \ D(x) \Rightarrow \neg L(x) \)
- \( \exists x \ D(x) \& I(x) \)

PROVE:

- \( \exists x \ I(x) \& \neg R(x) \)

Convert to CNF

- \( \forall x \ R(x) \Rightarrow L(x) \)
- \( \forall x \ D(x) \Rightarrow \neg L(x) \)
- \( \exists x \ D(x) \& I(x) \)

NEGATE CONCLUSION

- \( \neg \exists x \ I(x) \& \neg R(x) \)

\[ \neg R(x) \lor L(x) \]
\[ \neg D(y) \lor \neg L(y) \]
\[ D(A) \]
\[ I(A) \]
\[ \neg I(z) \lor R(z) \]
Resolution Is Powerful

- Runs Pretty Fast
- Lead to development of PROLOG
  - Used in Europe & Japan
  - We 'mercans like LISP!
But Still Has Problems

- Some Thought (in 1965) they solved satisfiability with a general purpose algorithm (resolution)
  - But this would mean P=NP
- Resolution Still Needs Heuristics:
  - Set of Support
  - Unit Preference
  - Human Knowledge