Propositional logic

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Simulated annealing competition

Top 3 simulated annealing entries for HW-3 Problem 2:
• Timothy Sweetser,
• Brian Taylor
• Rishi Sadhir
Logical inference problem

Logical inference problem:
• Given:
  – a knowledge base KB (a set of sentences) and
  – a sentence $\alpha$ (called a theorem),
• Does a KB semantically entail $\alpha$? $KB \models \alpha$?

In other words:
• In all interpretations in which sentences in the KB are true, is $\alpha$ also true?

Approaches to solve the logical inference problem:
• Truth-table approach
• Inference rules
• Conversion to SAT
  – Resolution refutation
Properties of inference solutions

- **Truth-table approach**
  - Blind
  - Exponential in the number of variables

- **Inference rules**
  - More efficient
  - Many inference rules to cover logic

- **Conversion to SAT - Resolution refutation**
  - More efficient
  - Sentences must be converted into CNF
  - One rule – the resolution rule - is sufficient to perform all inferences

KB in restricted forms

If the sentences in the KB are restricted to some special forms some of the sound inference rules may become complete

**Example:**

- **Horn form (Horn normal form)**
  - a clause with at most one positive literal
    
    \[(A \lor \neg B) \land (\neg A \lor \neg C \lor D)\]

  Can be written also as:

  \[(B \Rightarrow A) \land ((A \land C) \Rightarrow D)\]

- **Two inference rules that are sound and complete for KBs in the Horn normal form:**
  - Resolution
  - Modus ponens
KB in Horn form

• Horn form: a clause with at most one positive literal
  \((A \lor \neg B) \land (\neg A \lor \neg C \lor D)\)

• Not all sentences in propositional logic can be converted into the Horn form

• KB in Horn normal form:
  – Two types of propositional statements:
    • Rules
      \((\neg B_1 \lor \neg B_2 \lor \ldots \neg B_k \lor A)\)
      \(\equiv\)
      \((\neg (B_1 \land B_2 \land \ldots B_k) \lor A)\)
      \(\equiv\)
      \((B_1 \land B_2 \land \ldots B_k \Rightarrow A)\)
    • Propositional symbols: facts \(B\)

  – Application of the resolution rule:
    – Infers new facts from previous facts
      \[
      \frac{(A \lor \neg B), B}{A} \quad \frac{(A \lor \neg B), (B \lor \neg C)}{(A \lor C)}
      \]
    – Resolution is sound and complete for inferences on propositional symbols for KB in the Horn normal form (clausal form)

    • Similarly, modus ponens is sound and complete when the HNF is written in the implicative form
Complexity of inferences for KBs in HNF

Question:
How efficient the inferences in the HNF can be?
Answer:
Inference on propositional symbols \( \rightarrow \)
Procedures linear in the size of the KB in the Horn form exist.
• Size of a clause: the number of literals it contains.
• Size of the KB in the HNF: the sum of the sizes of its elements.
Example:
\[ A, B, (A \land B \Rightarrow C), (C \Rightarrow D), (C \Rightarrow E), (E \land F \Rightarrow G) \] or
\[ A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]
The size is: 12

How to do the inference? If the HNF (is in the clausal form) we can apply resolution.

\[ A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]
How to do the inference? If the HNF (is in the clausal form) we can apply resolution.

\[ A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]

Inferred facts: \( \neg B \lor C \)

Features:
- Every resolution is a positive unit resolution; that is, a resolution in which one clause is a positive unit clause (i.e., a proposition symbol).
Complexity of inferences for KBs in HNF

Features:
- At each resolution, the input clause which is not a unit clause is a logical consequence of the result of the resolution. (Thus, the input clause may be deleted upon completion of the resolution operation.)

\[ A, B, (\neg A \lor \neg B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]

\[ \neg B \lor C \]

\[ C \]

\[ D \]

\[ E \]
Complexity of inferences for KBs in HNF

Features:

- Following this deletion, the size of the KB (the sum of the lengths of the remaining clauses) is one less than it was before the operation.

\[ A, B, (\neg A \lor B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]

Complexity of inferences for KBs in HNF

Features:

- If \( n \) is the size of the KB, then at most \( n \) positive unit resolutions may be performed on it.

\[ A, B, (\neg A \lor B \lor C), (\neg C \lor D), (\neg C \lor E), (\neg E \lor \neg F \lor G) \]
**Complexity of inferences for KBs in HNF**

A linear time resolution algorithm:
- The number of positive unit resolutions is limited to the size of the formula (n)

- But to assure overall linear time we need to access each proposition in a constant time:
  - Data structures indexed by proposition names may be accessed in constant time. (This is possible if the proposition names are number in a range (e.g., 1..n), so that array lookup is the access operation.
  - If propositions are accessed by name, then a symbol table is necessary, and the algorithm will run in time $O(n \cdot \log(n))$.

---

**Forward and backward chaining**

Two inference procedures based on modus ponens for Horn KBs:
- **Forward chaining**
  - **Idea:** Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.

- **Backward chaining (goal reduction)**
  - **Idea:** To prove the fact that appears in the conclusion of a rule prove the premises of the rule. Continue recursively.

Both procedures are complete for KBs in the Horn form !!!
**Forward chaining example**

- **Forward chaining**
  
  **Idea:** Whenever the premises of a rule are satisfied, infer the conclusion. Continue with rules that became satisfied.

Assume the KB with the following rules and facts:

**KB:**

- **R1:** \( A \land B \Rightarrow C \)
- **R2:** \( C \land D \Rightarrow E \)
- **R3:** \( C \land F \Rightarrow G \)

**F1:** \( A \)
**F2:** \( B \)
**F3:** \( D \)

**Theorem:** \( E \)  

---

**Forward chaining example**

**Theorem:** \( E \)

**KB:**

- **R1:** \( A \land B \Rightarrow C \)
- **R2:** \( C \land D \Rightarrow E \)
- **R3:** \( C \land F \Rightarrow G \)

**F1:** \( A \)
**F2:** \( B \)
**F3:** \( D \)
Forward chaining example

Theorem: \( E \)

KB:  
R1: \( A \land B \Rightarrow C \)
R2: \( C \land D \Rightarrow E \)
R3: \( C \land F \Rightarrow G \)

F1: \( A \)
F2: \( B \)
F3: \( D \)

Rule R1 is satisfied.
F4: \( C \)

Rule R2 is satisfied.
F5: \( E \)
Forward chaining

• Efficient implementation: linear in the size of the KB
• Example:

\[
\begin{align*}
P & \Rightarrow Q \\
L \land M & \Rightarrow P \\
B \land L & \Rightarrow M \\
A \land P & \Rightarrow L \\
A \land B & \Rightarrow L \\
A & \\
B
\end{align*}
\]

Forward chaining

• Count the number of facts in the antecedent of the rule

\[
\begin{align*}
P & \Rightarrow Q \\
L \land M & \Rightarrow P \\
B \land L & \Rightarrow M \\
A \land P & \Rightarrow L \\
A \land B & \Rightarrow L \\
A & \\
B
\end{align*}
\]
Forward chaining

• Inferred facts decrease the count

\[ P \rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]

Forward chaining

• New facts can be inferred when the count associated with a rule becomes 0

\[ P \rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Forward chaining

- $P \Rightarrow Q$
- $L \land M \Rightarrow P$
- $B \land L \Rightarrow M$
- $A \land P \Rightarrow L$
- $A \land B \Rightarrow L$
- $A$
- $B$
Forward chaining

•

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Backward chaining example

- Backward chaining is more focused:
  - tries to prove the theorem only

KB:
- R1: \( A \land B \Rightarrow C \)
- R2: \( C \land D \Rightarrow E \)
- R3: \( C \land F \Rightarrow G \)
- F1: \( A \)
- F2: \( B \)
- F3: \( D \)
Backward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Backward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Backward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Backward chaining

\[ P \Rightarrow Q \quad \leftarrow \]
\[ L \land M \Rightarrow P \quad \leftarrow \]
\[ B \land L \Rightarrow M \quad \leftarrow \]
\[ A \land P \Rightarrow L \quad \leftarrow \]
\[ A \land B \Rightarrow L \quad \leftarrow \]
\[ A \quad \leftarrow \]
\[ B \quad \leftarrow \]
Backward chaining

\[ P \Rightarrow Q \]
\[ L \land M \Rightarrow P \]
\[ B \land L \Rightarrow M \]
\[ A \land P \Rightarrow L \]
\[ A \land B \Rightarrow L \]
\[ A \]
\[ B \]
Forward vs Backward chaining

- **FC is data-driven**, automatic, unconscious processing,
  - e.g., object recognition, routine decisions

- May do lots of work that is irrelevant to the goal

- **BC is goal-driven**, appropriate for problem-solving,
  - e.g., Where are my keys? How do I get into a PhD program?

- Complexity of BC can be **much less** than linear in size of KB

KB agents based on propositional logic

- Propositional logic allows us to build **knowledge-based agents** capable of answering queries about the world by inferring new facts from the known ones

- **Example**: an agent for diagnosis of a bacterial disease

  **Facts**:
  - The stain of the organism is gram-positive
  - The growth conformation of the organism is chains

  **Rules**: (If)
  - The stain of the organism is gram-positive ∧
  - The morphology of the organism is coccus ∧
  - The growth conformation of the organism is chains

  (Then) $\Rightarrow$ The identity of the organism is streptococcus
First-order logic

Limitations of propositional logic

World we want to represent and reason about consists of a number of objects with variety of properties and relations among them

Propositional logic:
- Represents statements about the world without reflecting this structure and without modeling these entities explicitly

Consequence:
- Some knowledge is hard or impossible to encode in the propositional logic.
- Two cases that are hard to represent:
  - Statements about similar objects, or relations
  - Statements referring to groups of objects.
Limitations of propositional logic

• **Statements about similar objects and relations needs to be enumerated**

• **Example:** Seniority of people domain
  For inferences we need:
  
  \( \text{John is older than Mary} \land \text{Mary is older than Paul} \rightarrow \text{John is older than Paul} \)
  
  \( \text{Jane is older than Mary} \land \text{Mary is older than Paul} \rightarrow \text{Jane is older than Paul} \)

• **Problem:** if we have many people and their age relations we need to represent many rules to support the inferences

• **Possible solution:** ??

For inferences we need:

\[ \text{Pers}_A \text{ is older than } \text{Pers}_B \land \text{Pers}_B \text{ is older than } \text{Pers}_C \rightarrow \text{Pers}_A \text{ is older than } \text{Pers}_C \]
Limitations of propositional logic

- **Statements referring to groups of objects require exhaustive enumeration of objects**
- **Example:**
  Assume we want to express *Every student likes vacation*

  Doing this in propositional logic would require to include statements about every student

  \[
  \text{John likes vacation} \land \text{Mary likes vacation} \land \text{Ann likes vacation} \land \cdots
  \]

  **Solution:** Allow quantification in statements

First-order logic (FOL)

- More expressive than **propositional logic**
- **Eliminates deficiencies of PL by:**
  - Representing objects, their properties, relations and statements about them;
  - Introducing variables that refer to arbitrary objects and can be substituted by a specific object
  - Introducing quantifiers allowing us to make statements over groups objects without the need to represent each of them separately