Clausal (Conjunctive Normal) Form and Resolution Techniques

Wednesday, 29 September 2004

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.kddresearch.org
http://www.cis.ksu.edu/~bhsu

Reading:
Chapter 9, Russell and Norvig
Handout, Nilsson and Genesereth
• **Today’s Reading**
  – Chapter 9, Russell and Norvig
  – Recommended references: Nilsson and Genesereth (excerpt of Chapter 5 online)

• **Thursday’s Reading:** Chapter 9, R&N

• **Previously:** Propositional and First-Order Logic
  – Two weeks ago
    • Logical agents: KR, inference, problem solving
    • Propositional logic: normal forms, sequent rules
    • Predicates and terms
    • First-order logic (FOL): quantifiers
  – Last week
    • FOL agents; frame problem; situation calculus, successor-state axioms
    • FOL KBs and forward search using sequent rules (sound but incomplete set)

• **Today:** Backward Inference
  – Resolution refutation (sound and complete proof procedure)
  – Computability (decidability) issues
Review: Automated Deduction by Forward Chaining

Operators are inference rules
States are sets of sentences
Goal test checks state to see if it contains query sentence

AI, UE, MP is a common inference pattern

Problem: branching factor huge, esp. for UE

Idea: find a substitution that makes the rule premise match some known facts
⇒ a single, more powerful inference rule

\[
\frac{\alpha, \alpha \Rightarrow \beta}{\beta}
\]

\[
\frac{\alpha \land \beta}{\alpha \land \beta}
\]

\[
\frac{\forall x \alpha}{\alpha(x/\tau)}
\]

- Modus Ponens
- And Introduction
- Universal Elimination

Adapted from slides by S. Russell, UC Berkeley
Conjunctive Normal (aka Clausal) Form [2]: Conversion (Nilsson) and Mnemonic

- Implications Out
- Negations Out
- Standardize Variables Apart
- Existentials Out (Skolemize)
- Universals Made Implicit
- Distribute And Over Or (i.e., Disjunctions In)
- Operators Out
- Rename Variables
- A Memonic for Star Trek: The Next Generation Fans

Captain Picard:

I’ll Notify Spock’s Eminent Underground Dissidents On Romulus

I’ll Notify Sarek’s Eminent Underground Descendant On Romulus
Offline Exercise: 
**Read-and-Explain Pairs**

- For Class Participation (PS3, MP4)
- With Your Term Project Partner or Assigned Partner(s)

- Read: Chapter 9 (esp. 9.2, 9.5), Chapter 10 R&N 2e
- By Fri 08 Oct 2004, Fri 15 Oct 2004
Offline Exercise: 
**Read-and-Explain Pairs**

- **For Class Participation (MP4)**
- **With Your Term Project Partner or Assigned Partner(s)**
  - Read your assigned sections (*take notes* if needed)
    - Group A: R&N Sections 9.3, 9.6 p. 284-286, 10.2 p. 302-303, 10.4
    - Group B: R&N Sections 9.7, 10.1, 10.2 p. 299-302, 10.3 p. 304-305, 10.5-10.8
  - Skim your partner’s sections
  - Meet with your partner (by e-mail, ICQ, IRC, or in person)
  - Explain your section
    - Key ideas – what’s important?
    - Important technical points
  - Discuss unclear points and *write them down*!
- **By Fri 08 Oct 2004**
  - Post
    - Confirmation to ksu-cis730-fall2004
    - **Muddiest point**: what is least clear in your understanding of your section?
  - Re-read your partner’s section as needed
Review:
Logic Programming (Prolog) Examples

Depth-first search from a start state X:

dfs(X) :- goal(X).
dfs(X) :- successor(X,S), dfs(S).

No need to loop over S: successor succeeds for each

Appending two lists to produce a third:

append([],Y,Y).
append([X|L],Y,[X|Z]) :- append(L,Y,Z).

query: append(A,B,[1,2]) ?
answers: A=[]   B=[1,2]
         A=[1,2] B=[]
Completeness of Resolution

- Any Set of Sentences $S$ is Representable in Clausal Form (Last Class)
- Assume $S$ is Unsatisfiable, and in Clausal Form
- (By Herbrand’s Theorem) Some Set $S'$ of Ground Instances is Unsatisfiable
- (By Ground Resolution Theorem) Resolution Derives $\bot$ From $S'$
- (By Lifting Lemma) $\exists$ A Resolution Proof $S \vdash \bot$

Figure 9.13 p. 301 R&N 2e
Decidability Revisited

- See: Section 9.7 Sidebar, p. 288 R&N
- Duals (Why?)

\[
\begin{align*}
L_{\text{VALID}} & \quad \overline{L_{\text{VALID}}} \\
L_{\text{SAT}} & \quad L_{\text{SAT}}
\end{align*}
\]

- Complexity Classes

- Understand: Reduction to \( L_d, L_H \)
Unification Procedure:
General Idea

A substitution $\sigma$ unifies atomic sentences $p$ and $q$ if $p\sigma = q\sigma$

<table>
<thead>
<tr>
<th>$p$</th>
<th>$q$</th>
<th>$\sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Knows}(\text{John}, x)$</td>
<td>$\text{Knows}(\text{John}, \text{Jane})$</td>
<td>${x/\text{Jane}}$</td>
</tr>
<tr>
<td>$\text{Knows}(\text{John}, x)$</td>
<td>$\text{Knows}(y, \text{OJ})$</td>
<td>${x/\text{John}, y/\text{OJ}}$</td>
</tr>
<tr>
<td>$\text{Knows}(\text{John}, x)$</td>
<td>$\text{Knows}(y, \text{Mother}(y))$</td>
<td>${y/\text{John}, x/\text{Mother}(\text{John})}$</td>
</tr>
</tbody>
</table>

**Idea:** Unify rule premises with known facts, apply unifier to conclusion
E.g., if we know $q$ and $\text{Knows}(\text{John}, x) \Rightarrow \text{Likes}(\text{John}, x)$
then we conclude
- $\text{Likes}(\text{John}, \text{Jane})$
- $\text{Likes}(\text{John}, \text{OJ})$
- $\text{Likes}(\text{John}, \text{Mother}(\text{John}))$

- **Most General Unifier** (Least-Commitment Substitution)
- See: Examples (p. 271 R&N, Nilsson and Genesereth)

Adapted from slides by S. Russell, UC Berkeley

CIS 730: Introduction to Artificial Intelligence
Logic Programming – Tricks of The Trade [1]: Dealing with Equality

- **Problem**
  - How to find appropriate inference rules for sentences with =?
  - Unification OK without it, but…
  - $A = B$ doesn’t force $P(A)$ and $P(B)$ to unify

- **Solutions**
  - **Demodulation**
    - *Generate substitution from equality term*
    - Additional sequent rule: p. 284 R&N
  - **Paramodulation**
    - More powerful
    - *Generate substitution from WFF containing equality constraint*
    - e.g., $(x = y) \lor P(x)$
    - Sequent rule sketch: p. 284 R&N
Logic Programming – Tricks of The Trade [2]: Resolution Strategies

- **Unit Preference**
  - Idea: Prefer inferences that produce shorter sentences (compare: Occam’s Razor)
  - How? Prefer unit clause (single-literal) resolvents
  - Reason: trying to produce a short sentence ($\perp \equiv \text{True} \Rightarrow \text{False}$)

- **Set of Support**
  - Idea: try to eliminate some potential resolutions (prevention as opposed to cure)
  - How? Maintain set SoS of resolution results and always take one resolvent from it
  - Caveat: need right choice for SoS to ensure completeness

- **Input Resolution and Linear Resolution**
  - Idea: “diagonal” proof (proof “list” instead of proof tree)
  - How? Every resolution combines some input sentence with some other sentence
  - Input sentence: in original KB or query
  - Generalize to linear resolution: include any ancestor in proof tree to be used

- **Subsumption**
  - Idea: eliminate sentences that sentences that are more specific than others
  - E.g., $P(x)$ subsumes $P(A)$
Logic Programming – Tricks of The Trade [3]:
Indexing Strategies

• Store and Fetch
  – Idea: store knowledge base in list of conjuncts
  – STORE: constant, i.e., $O(1)$ worst-case running time
  – FETCH: linear, i.e., $O(n)$ time

• Table Based
  – Idea: store KB in hash table (key: ground literals)
  – STORE: $O(1)$
  – FETCH: $O(1)$ expected case
  – Problems
    • Complex WFFs (other than negated atoms)
    • Variables
      – Solution: implicative normal form matching (Figure 10.1, p. 301 R&N)

• Tree-Based
  – What if there are many clauses for a predicate? (e.g., Brother (012-34-5678, x))
  – Type of combined indexing: joint primary key – predicate and argument symbols
  – May need background knowledge for semantic query optimization (SQO)
Logic Programming – Tricks of The Trade [4]: Compilation

- Intermediate Languages
  - Abstract machines
    - Warren Abstract Machine (WAM)
    - Java Virtual Machine (JVM)
  - Imperative intermediate representations (IRs)
    - C/C++
    - LISP / Scheme / SML – functional languages with imperative features
- Use in Genetic Programming (GLP): Later
- Beyond Scope of CIS 730: Compiling with Continuations (Appel)

Adapted from slides by S. Russell, UC Berkeley
Summary Points

• Previously: FOL, Forward and Backward Chaining, Resolution
• Today: More Resolution Theorem Proving, Prolog, and Unification
  – Review: resolution inference rule
    • Single-resolvent form
    • General form
  – Application to logic programming
  – Review: decidability properties
    • FOL-SAT
    • FOL-NOT-SAT (language of unsatisfiable sentences; complement of FOL-SAT)
    • FOL-VALID
    • FOL-NOT-VALID
  – Unification
• Next Week
  – Intro to classical planning
  – Inference as basis of planning
Terminology

- Properties of Knowledge Bases (KBs)
  - Satisfiability and validity
  - Entailment and provability
- Properties of Proof Systems
  - Soundness and completeness
  - Decidability, semi-decidability, undecidability
- Resolution
- Refutation
- Satisfiability, Validity
- Unification
  - Occurs check
  - Most General Unifier
- Prolog: Tricks of The Trade
  - Demodulation, paramodulation
  - Unit resolution, set of support, input / linear resolution, subsumption
  - Indexing (table-based, tree-based)