Resolution

71

Resolution

- An application of a special form of Modus Ponens.
- Can also be considered a form of backward chaining.
- Provides a method for automated theorem proving.
- Assumes all sentences are in conjunctive normal form (a.k.a. clause form)
  - $(p_1 \vee q_1 \vee r_1) \wedge (p_2 \vee q_2 \vee r_2) \wedge (p_3 \vee q_3 \vee r_3)$

72

Converting to Clause Form

- Eliminate implications
- Reduce scope of negation symbols
- Standardize variables
- Eliminate existential quantifiers
- Move universal quantifiers left
- Rewrite in conjunctive normal form
- Eliminate universal quantifiers
- Separate into clauses
- Rename variables

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

73

A Detailed Example

- Convert the following to clause form:
  - $\forall x\ [P(x) \Rightarrow [\forall y\ [P(y) \Rightarrow P(f(x,y))] \wedge \neg\forall y\ [Q(x,y) \Rightarrow P(y)]]]$
  - where P and Q are predicates, x and y are variables, and f is a function

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

74

Step 1

- Eliminate implications
- Recall $x \Rightarrow y$ is the same as $\neg x \lor y$
- Conversion yields
  - $\forall x\ [\neg P(x) \lor [\forall y\ [\neg P(y) \lor P(f(x,y))] \land \neg\forall y\ [\neg Q(x,y) \lor P(y)]]]$

MONTANA STATE UNIVERSITY | Norm Asbjornson College of Engineering

75

Step 2

- Reduce scope of negation symbols
- DeMorgan's laws
  - $\neg(X \land Y) \equiv \neg X \lor \neg Y$
  - $\neg(X \lor Y) \equiv \neg X \land \neg Y$
- Double negation: $\neg(\neg X) \equiv X$
- Duality of quantifiers
  - $\neg\forall x\ P(x) \equiv \exists x\ [\neg P(x)]$
  - $\neg\exists x\ P(x) \equiv \forall x\ [\neg P(x)]$

MONTANA STATE UNIVERSITY | Norm Asbjornson College of Engineering

76

Step 2 (cont.)

- Only part of sentence affected:
  - $\neg\forall y\ [\neg Q(x,y) \vee P(y)]$
- Apply duality of quantifiers:
  - $\exists y\ [\neg[\neg Q(x,y) \vee P(y)]]$
- Apply DeMorgan's:
  - $\exists y\ [\neg[\neg Q(x,y)] \wedge \neg P(y)]$
- Apply double negation:
  - $\exists y\ [Q(x,y) \wedge \neg P(y)]$

Step 3

- Standardize variables
- Results in renaming variables associated with each quantifier to be unique.
- Conversion yields
  - $\forall x\ [\neg P(x) \vee [\forall y\ [\neg P(y) \vee P(f(x,y))] \wedge \exists w\ [Q(x,w) \wedge \neg P(w)]]]$
- Be careful of scope of quantifiers!!

Step 4

- Eliminate existential quantifiers
- Skolem constants
  - Replace existentially quantified variables by a unique constant.
- Skolem functions
  - If existentially quantified variable is within scope of a universal quantifier too, replace variables with a function of universally quantified variable.

79

Step 4 (cont.)

- Conversion yields
  - $\forall x\ [\neg P(x) \vee [\forall y\ [\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]]]$
- Note the Skolemization did not include $\forall y$ since the brackets limit the scope of this quantifier.

80

Step 5

- Move all universal quantifiers left.
- This is called *prenex* form.
- This is allowed because all variables are unique to each universal quantifier.
- Conversion yields
  - $\forall x\ \forall y\ [\neg P(x) \vee [\neg P(y) \vee P(f(x,y))] \wedge [Q(x,g(x)) \wedge \neg P(g(x))]]$

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

81

Step 6

- Rewrite in Conjunctive Normal Form (CNF)
- Uses distributive law twice:
  - $A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
- Conversion yields
  - $\forall x\ \forall y\ [[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]]$

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

82

Step 7

- Eliminate universal quantifiers
- From here on, all variables are *assumed* to be universally quantified.
- Conversion yields
  - $[[\neg P(x) \vee \neg P(y) \vee P(f(x,y))] \wedge [\neg P(x) \vee Q(x,g(x))] \wedge [\neg P(x) \vee \neg P(g(x))]]$

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

83

Step 8

- Separate into clauses
- This amounts to AND-elimination
- Conversion yields
  - $\neg P(x) \vee \neg P(y) \vee P(f(x,y))$
  - $\neg P(x) \vee Q(x,g(x))$
  - $\neg P(x) \vee \neg P(g(x))$

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

84

Step 9

- Rename variables
- No variable symbol should appear in more than one clause
- Conversion yields
  - $\neg P(x_1) \vee \neg P(y) \vee P(f(x_1,y))$
  - $\neg P(x_2) \vee Q(x_2,g(x_2))$
  - $\neg P(x_3) \vee \neg P(g(x_3))$

Resolution Theorem Proving

- Proof by contradiction
- Unification used to match terms in clauses
- Procedure:
  - Negate the theorem to be proven
  - Add to axiom set
  - Convert all axioms to clause form
  - Resolve clauses until either the empty clause is produced or no resolvable clauses remain
  - Empty clause proves theorem (contradiction)

Detailed Example

- Given
  - Whoever can read is literate.
  - Dolphins are not literate.
  - Some dolphins are intelligent.
- Prove
  - Some who are intelligent cannot read.

87

Convert to FOL

- Given
  - $\forall x\ [Read(x) \Rightarrow Literate(x)]$
  - $\forall x\ [Dolphin(x) \Rightarrow \neg Literate(x)]$
  - $\exists x\ [Dolphin(x) \wedge Intelligent(x)]$
- Prove
  - $\exists x\ [Intelligent(x) \wedge \neg Read(x)]$

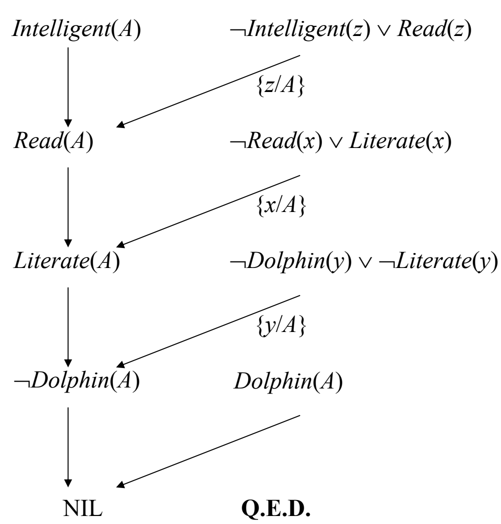
88

Convert to Clause Form

- Axioms
  - $\neg Read(x) \vee Literate(x)$
  - $\neg Dolphin(y) \vee \neg Literate(y)$
  - $Dolphin(A)$ {A is a Skolem constant)
  - $Intelligent(A)$
- Negated Theorem
  - $\neg\exists x\ [Intelligent(x) \wedge \neg Read(x)]$, which yields in clause form
  - $\neg Intelligent(z) \vee Read(z)$

89

Resolution Proof Tree



90

Resolution Search Strategies

- Unit Preference
  - Always try to resolve with single literals
- Set of support
  - Start with negated query and only resolve against descendents of that query
- Input Resolution
  - Every resolution combines an input sentence (KB or query) with some other sentence
  - Linear resolution is a slight generalization
- Subsumption
  - Only keep the most general set of sentences around

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

91

Why is Resolution Complete

- Completeness says, "if it is true, we can derive it."
- Any set of sentences can be put into an equivalent clausal form
- Assume S is in clausal form and unsatisfiable
- Some set S' of ground clauses generated from S is unsatisfiable
- Resolution can find the contradiction in S'
- By a "lifting lemma," we can take this ground proof and put back the variables to get the refutation of the original sentences

MONTANA STATE UNIVERSITY | NORM ASBJORNSON College of ENGINEERING

92