16.1 Unification

- Abbreviating wffs of the form \((\forall \xi_1, \xi_2, \ldots, \xi_n)(\lambda_1 \lor \lambda_2 \lor \cdots \lor \lambda_k)\) by \(\lambda_1 \lor \lambda_2 \lor \cdots \lor \lambda_k\), where \(\lambda_1, \lambda_2, \ldots, \lambda_k\) are literals that might contain occurrences of the variables \(\xi_1, \xi_2, \ldots, \xi_n\).
- Simply dropping the universal quantifiers and assuming universal quantification of any variables in the \(\lambda_i\).
- *Clauses*: WFFs in the abbreviated form.
- If two clauses have matching but complementary literals, it is possible to resolve them.
  - Example: \(P(f(y), A) \lor Q(B, C)\) vs. \(\neg P(x, A) \lor R(x, C) \lor S(A, B)\)

\[
P(f(y), A) \quad \neg P(f(y), A) \lor R(f(y), C) \lor S(A, B)
\]

\[
R(f(y), C) \lor S(A, B) \lor Q(B, C)
\]
16.1 Unification

Any substitution can be represented by a set of ordered pairs
\[ S = \{ (x_1/\xi_1, x_2/\xi_2, \ldots, x_n/\xi_n) \} \]
- The pair \( x_i/\xi_i \) means that term \( x_i \) is substituted for every occurrence of the variable \( \xi_i \) throughout the scope of the substitution.
- No variables can be replaced by a term containing that same variable.
- The substitutions used earlier in obtaining the four instances of \( P(x, f(y), B) \)
  \[ s_1 = (x/x, y/y) \]
  \[ s_2 = (\xi_1/\eta_1, \xi_2/\eta_2) \]
  \[ s_3 = (\xi_1/\eta_1, \eta_2/\xi_2) \]
  \[ s_4 = (\xi_1/\eta_1, \xi_2/\eta_2) \]
  \[ w' \text{ denotes a substitution instance of an expression } w, \text{ using a substitution } s \]
  \[ P(x, f(y), B) = f(x, f(y), B) \]
  The composition \( s_1 \) and \( s_2 \) is denoted by \( s_1 \circ s_2 \), which is that substitution obtained by first applying \( s_2 \) to the terms of \( s_1 \) and then adding any pairs of \( s_2 \) having variables not occurring among the variables of \( s_1 \). Thus,
  \[ (g(x, y) / z)(3, 4)/B(y, C(w), D(x, y)) = (g(A, B) / z, A(x, B) / y, C(w)) \]

(C) 2000 SNU CSE Artificial Intelligence Lab (SCAI)

16.1 Unification

- MGU (Most general (or simplest) unifier) has the property that if \( s \) is any unifier of \( \{\omega\} \) yielding \( \{\omega\} \subset s \), then there exists a substitution \( \phi \) such that \( \{\omega\} = \{\omega\} \subset \phi \). Furthermore, the common instance produced by a most general unifier is unique except for alphabetic variants.
- UNIFY
  - Can find the most general unifier of a finite set of unifiable expressions and that report failure when the set cannot be unified.
  - Works on a set of list-structured expressions in which each literal and each term is written as a list.
  - Basic to UNIFY is the idea of a disagreement set. The disagreement set of a nonempty set \( W \) of expressions is obtained by locating the first symbol at which not all the expressions in \( W \) have exactly the same symbol, and then extracting from each expression in \( W \) the subexpression that begins with the symbol occupying that position.

(C) 2000 SNU CSE Artificial Intelligence Lab (SCAI)

16.1 Unification

- (\( ax \)) = (\( ax, y \)), (\( ax, y \)) = (\( ax, y \))
- Let \( \omega \) be \( P(x, y) \), \( \sigma \) be \( \{f(y)/x\} \), and \( \tau \) be \( \{A/y\} \) then,
  \[ (\omega x) = [P(f(y), A)](A/y) = P(f(A), A) \]
  and
  \[ \sigma(\omega x) = [P(x, y)](A/y) = P(f(A), A) \]
- Substitutions are not, in general, commutative
  \[ \sigma(\omega x) = [P(x, y)](A/y) = P(f(A), A) \]
- Unifiable: a set of \( \{\omega\} \) expressions is unifiable if there exists a substitution \( s \) such that \( \omega = \omega \subset s = \omega \subset s \ldots \)
- \( (A / x, B / y) \) unifies \( (P(x, f(y), B), P(x, f(B), B)) \) to yield \( (P(x, f(A), B)) \)

(C) 2000 SNU CSE Artificial Intelligence Lab (SCAI)
16.2 Predicate-Calculus Resolution

- \( \gamma_1, \gamma_2 \) are two clauses. Atom \( \theta \) in \( \gamma_1 \) and a literal \( \neg \phi \) in \( \gamma_2 \) such that \( \theta \) and \( \phi \) have a most general unifier, \( \mu \), then these two clauses have a resolvent, \( \rho \). The resolvent is obtained by applying the substitution \( \mu \) to the union of \( \gamma_1 \) and \( \gamma_2 \), leaving out the complementary literals. \( \rho \equiv (\gamma_1 - \{ \theta \}) \cup (\gamma_2 - \{ \neg \phi \}) \mu \)

- Examples:

\[
\begin{align*}
&\{ P(x) \}, \{ Q(x, y) \}, \{ \neg P (A), R (B, x) \} \rightarrow \{ Q (A), y, R (B, x) \} \\
&\{ P(x, z), Q(x, x) \}, \{ \neg P (A, z), R (B, x) \} \rightarrow \{ Q (A, y), R (B, x), \neg P (A, z) \}
\end{align*}
\]

16.3 Completeness and Soundness

- Predicate-calculus resolution is sound
  - If \( \theta \) is the resolvent of two clauses \( \phi \) and \( \psi \), then \( \{ \phi, \psi \} = \rho \)
- Completeness of resolution
  - It is impossible to infer by resolution alone all the formulas that are logically entailed by a given set.
  - In propositional resolution, this difficulty is surmounted by using resolution refutation.

16.4 Converting Arbitrary wffs to Clause Form

1. Eliminate implication signs.
2. Reduce scopes of negation signs.
3. Standardize variables
   - Since variables within the scopes of quantifiers are like “dummy variables”, they can be renamed so that each quantifier has its own variable symbol.
   
   \( (\forall x)[(\exists y) \neg P(x) \lor (\exists x) Q(x)] \rightarrow (\forall x)[(\exists y) \neg P(x) \lor (\exists x) Q(y)] \)

4. Eliminate existential quantifiers.

16.4 Converting Arbitrary wffs to Clause Form

- Skolem function, Skolemization: 
  
  \[
  (\forall x)[(\exists y) \text{Height}(x, y)] \\
  (\forall x) \text{Height}[x, \kappa(x)]
  \]

- Replace each occurrence of its existentially quantified variable by a Skolem function whose arguments are those universally quantified variables.

- Function symbols used in Skolem functions must be “new”.

\[
\begin{align*}
&\{ (\forall w)[Q(w)] \} \supset \{ (\forall x)[(\exists z)[P(x, y, z) \supset (\forall w)[R(x, y, u, z)]]] \}
\rightarrow &\{ (\forall w)[Q(w)] \} \supset \{ (\forall x)[(\exists z)[P(x, y, g, y) \supset (\forall w)[R(x, y, u, z)]]] \}
\rightarrow &\{ (\forall x)[(\exists z)[P(x, y) \lor (\forall y)[\neg P(x) \lor P(x, y)]] \supset (\exists z)[Q(x, w) \land \neg P(w)]] \}
\rightarrow &\{ (\forall x)[(\exists z)[P(x, y) \lor (\forall y)[\neg P(y) \lor P(x, y)]] \supset (\exists z)[Q(x, w) \land \neg P(w)]] \}
\rightarrow &\{ (\forall x)[(\exists z)[P(x, y) \lor (\forall y)[\neg P(y) \lor P(x, y)]] \supset (\exists z)[Q(x, w) \land \neg P(w)]] \}
\end{align*}
\]
16.4 Converting Arbitrary wffs to Clause Form

- Skolem function of no arguments
  \( (\exists x) P(x) \Rightarrow P(\overline{s}) \)

- Skolem form: To eliminate all of the existentially quantified variables from a wff, the preceding procedure on each subformula is used in turn. Eliminating the existential quantifiers from a set of wffs produces what is called the Skolem form of the set of formulas.

- The skolem form of a wff is not equivalent to the original wff. 
  
  \[ P(A \lor P(B)) \land (\exists x) P(x), \text{ but } [P(A) \lor P(B)] \land P(\overline{s}). \]

  What is true is that a set of formulas, \( \Delta \) is satisfiable if and only if the Skolem form of \( \Delta \) is unsatisfiable.

5. Convert to prenex form

- At this stage, there are no remaining existential quantifiers, and each universal quantifier has its own variable symbol.

- A wff in prenex form consists of a string of quantifiers called a prefix followed by a quantifier-free formula called a matrix. The prenex form for the example wff marked with an * earlier is

\[ \forall x(\forall y)(\neg P(x) \lor \neg Q(x,y) \lor P(x,y) \lor Q(x, \overline{h}(x))) \]

6. Put the matrix in conjunctive normal form

- When the matrix of the preceding example wff is put in conjunctive normal form, it became

\[ \forall x(\forall y)(\neg P(x) \lor \neg P(x,y) \lor P(x,y) \lor Q(x, \overline{h}(x))) \]

7. Eliminate universal quantifiers

- Assume that all variables in the matrix are universally quantified.

8. Eliminate \( \land \) symbols

- The explicit occurrence of \( \land \) symbols may be eliminated by replacing expressions of the form \( (a_1 \land a_2) \) with the set of wffs \( \{a_1, a_2\} \).

\[ \neg P(x) \lor \neg P(y) \lor P(\overline{h}(x, y)) \]

\[ \neg P(x) \lor Q(x, \overline{h}(x)) \]

\[ \neg P(x) \lor P(\overline{h}(x)) \]

(C) 2000 SNU CSE Artificial Intelligence Lab (SCAI)
16.5 Using Resolution to Prove Theorem

- To prove \( \omega \) from \( \Delta \), proceed just as in the propositional calculus. Negate \( \omega \), convert this negation to clause form, and add it to the clause form of \( \Delta \). Then apply resolution until the empty clause is deduced.

- Problem: the package delivery robot. Suppose this robot knows that all of the packages in room 27 are smaller than any of the ones in room 28.

  1. \( \forall x, y, z \) \( \text{Package}(x) \land \text{Package}(y) \land \text{Large}(x, y) \rightarrow \text{Small}(x, y) \)

   Abbreviating the predicate symbols to make our formulas more compact and converting to clause from yields:

  2. \( \neg \text{Large}(x, y) \lor \neg \text{Large}(y, x) \lor \text{Large}(y, 28) \lor \text{Large}(x, 28) \)

   Suppose that the robot knows that package A is either in room 27 or in room 28 (but not which). It knows that package B is in room 27 and that package B is not smaller than package A.

- 3. \( \text{P}(A) \)
- 4. \( \text{P}(B) \)
- 5. \( \text{I}(A, 27) \lor \text{I}(A, 28) \)
- 6. \( \text{I}(B, 27) \)
- 7. \( \neg \text{S}(B, A) \)

16.6 Answer Extraction

16.7 The Equality Predicate

- The relation constants used in the formulas in a knowledge base usually have intended meanings, but these relations are circumscribed only by the set of models of the knowledge base and not at all by the particular symbols used for relation constants. The result of resolution refutations will be consistent with intended meanings only if the knowledge base suitably constrains the actual relations.

- Equality relation: Equals(A,B) or A=B
  - Reflexive (\( \forall x \) Equals(x,x))
  - Symmetric (\( \forall x, y \)(Equals(x, y) \( \Rightarrow \) Equals(y, x)))
  - Transitive (\( \forall x, y, z \)(Equals(x, y) \( \land \) Equals(y, z) \( \Rightarrow \) Equals(x, z)))
16.7 The Equality Predicate

- **Paramodulation**
  - Equality-specific inference rule to be used in combination with resolution in cases where the knowledge base contains the equality predicate.
  - γ, γ are two clauses. If \( \gamma_1 = \{ \tau(\xi) \lor \gamma_2 \} \) and \( \gamma_2 = \{ \text{Eq}(\alpha, \beta) \lor \gamma_3 \} \), where \( \tau \), \( \alpha, \beta \) are terms, where \( \gamma_1 \) are clauses, and where \( \lambda(\xi) \) is a literal containing the term \( \tau \), and if \( \tau \) and \( \alpha \) have a most general unifier \( \sigma \), then infer the binary paramodulant of \( \gamma \) and \( \gamma_3 \): 
    \[
    \gamma' = \{ \lambda(\xi) \lor \gamma_3 \} \text{ where } \lambda(\xi) = \lambda(\xi)\lceil (\beta/\alpha) \rceil
    \]
  - Prove \( P(B) \) from \( P(A) \) and \( A = B \)
    - For a resolution-style proof, we must deduce the empty clause from the clauses \( \neg P(B), P(A), \) and \( A = B \).
    - Using paramodulation on the last two clauses, \( \lambda(\xi) \) is \( P(A) \), \( \tau \) is \( \alpha \), \( \alpha \) is \( A \), and \( \beta \) is \( B \). Since A (in the role of \( \tau \)) and A (in the role of \( \alpha \)) unify trivially without a substitution, the binary paramodulation is \( P(B) \), which is the result of replacing an occurrence of \( \tau \) (that is A) with \( \beta \) (that is B). Resolving this paramodulant with \( \neg P(B) \) yields the empty clause.

(C) 2000 SNU CSE Artificial Intelligence Lab (SCAI) 21

---

Additional Readings and Discussion

- Some people find the resolution inference rule unintuitive and prefer so-called natural-deduction methods. These are called “natural” because inference is performed on sentences more or less “as is” without transformations into canonical forms.
- Predicate evaluation is an instance of a more general process called *semantic attachment* in which data structure and programs are associated with elements of the predicate-calculus language. Attached structures and procedures can then be used to evaluate expressions in the language in a way that corresponds to their intended interpretations.