Lecture 16

Resolution for Predicate Logic

Instructor: Yu Zhen Xie
Revisit: main rules of inference in propositional logic

• Valid argument: 
  AND of premises → conclusion is a tautology

• Modus ponens:
  \((p \to q) \land p \to q\) is a tautology

• Hypothetical syllogism:
  \((p \to q) \land (q \to r) \to (p \to r)\) is a tautology

• Disjunctive syllogism:
  \((A \lor B) \land \neg A \to B\) is a tautology

• Resolution:
  \((A \lor C) \land (B \lor \neg C) \to (A \lor B)\) is a tautology
Rules of inference

• These patterns describe how new knowledge can be derived from existing knowledge, both in the form of propositional logic formulas (sentences).

• When describing an inference rule, the *premise* specifies the pattern that must match our knowledge base and the *conclusion* is the new knowledge inferred.
Modus ponens, modus tollens, AND elimination, AND introduction, and universal instantiation

• If the sentences \( P \) and \( P \rightarrow Q \) are known to be true, then **modus ponens** lets us infer \( Q \).

• Under the inference rule **modus tollens**, if \( P \rightarrow Q \) is known to be true and \( Q \) is known to be false, we can infer \( P \).

• **AND elimination** allows us to infer the truth of either of the conjuncts from the truth of a conjunctive sentence. E.g. \( P \land Q \) lets conclude both \( P \) and \( Q \) are true.

• **AND introduction** lets us infer the truth of a conjunction from the truth of its conjuncts. E.g. if both \( P \) and \( Q \) are true, then \( P \land Q \) are true.

• **Universal instantiation** states that if any universally quantified variable in a true sentence is replaced by any appropriate term from the domain, the result is a true sentence. Thus, if \( a \) is from the domain of \( X \), \( \forall X \ p(X) \) lets us infer \( p(a) \).
Definition

- A predicate logic (or calculus) expression $X$ **logically follows** from a set $S$ of predicate calculus expressions if every interpretation and variable assignment that satisfies $S$ also satisfies $X$.
  - An *interpretation* is an assignment of specific values to domains and predicates.

- An inference rule is **sound** if every predicate calculus expressions also logically follows from $S$.

- An inference rule is **complete** if, given a set $S$ of predicate calculus expressions, the rule can infer every expression that logically follows from $S$. 
Logic and finding a proof

• Given
  – a knowledge base represented as a set of propositional sentences.
  – a goal stated as a propositional sentence
  – list of inference rules

• We can write a program to repeatedly apply inference rules to the knowledge base in the hope of deriving the goal.
Developing a proof procedure

• Deriving (or refuting) a goal from a collection of logic facts corresponds to a very large search tree.

• A large number of rules of inference could be utilized.

• The selection of which rules to apply and when would itself be non-trivial.
Resolution and CNF

• *Resolution* is a single rule of inference that can operate efficiently on a special form of sentences.

• The special form is called *conjunctive normal form* (CNF) or *clausal form*, and has these properties:
  – Every sentence is a disjunction (OR) of literals (clauses)
  – All sentences are implicitly conjuncted (ANDed).
We have to worry about the arguments to predicates, so it is harder to know when two literals match and can be used by resolution.

– For example, does the literal \( \text{Father}(\text{Bill, Chelsea}) \) match \( \text{Father}(x, y) \)?

The answer depends on how we substitute values for variables.
Proof procedure for predicate logic

• Same idea, but a few added complexities:
  – conversion to CNF is much more complex.
  – Matching of literals requires providing a matching of variables, constants and/or functions.

\[ \neg \text{Skates}(x) \lor \text{LikesHockey}(x) \]
\[ \neg \text{LikesHockey}(y) \]

We can resolve these only if we assume \( x \) and \( y \) refer to the same object.
Predicate Logic and CNF

- Converting to CNF is harder - we need to worry about variables and quantifiers.
  - Eliminate all implications →
  - Reduce the scope of all → to single term
  - Make all variable names unique
  - Move quantifiers left (prenex normal form)
  - Eliminate Existential Quantifiers
  - Eliminate Universal Quantifiers
  - Convert to conjunction of disjuncts
  - Create separate clause for each conjunct.
Eliminate Existential Quantifiers

• Any variable that is existentially quantified means that
  – *there is some value for that variable that makes the expression true.*

• To eliminate the quantifier, we can replace the variable with a function.

• We don’t know what the function is, we just know it exists.
Skolem functions

• Named after the Norwegian logician Thoralf Skolem

• **Example:** $\exists y \text{ President}(y)$  
  We replace $y$ with a new function $\text{func}$:
  President($\text{func}()$)
  $\text{func}$ is called a Skolem function.

• In general the function must have the same number of arguments as the number of universal quantifiers in the current scope.
Skolemization Example

• In general the function must have the same number of arguments as the number of universal quantifiers in the current scope.

• **Example:** $\forall x \exists y \text{Father}(y, x)$
  - create a new function named *foo* and replace *y* with the function.
  - $\forall x \text{Father}(\text{foo}(x), x)$
Unification

• Two formulas are said to unify if there are legal instantiations (assignments of terms to variables) that make the formulas in question identical.

• The act of unifying is called unification. The instantiation that unifies the formulas in question is called a unifier.

• There is a simple algorithm called the unification algorithm that does this.
Unification

- **Example**: Unify the formulas $Q(a, y, z)$ and $Q(y, b, c)$

- **Solution**:
  - Since $y$ in $Q(a, y, z)$ is a different variable than $y$ in $Q(y, b, c)$, rename $y$ in the second formula to become $y1$.
  - This means that one must unify $Q(a, y, z)$ with $Q(y1, b, c)$.
  - An instance of $Q(a, y, z)$ is $Q(a, b, c)$ and an instance of $Q(y1, b, c)$ is $Q(a, b, c)$.
  - Since these two instances are identical, $Q(a, y, z)$ and $Q(y, b, c)$ unify.
  - The unifier is $y1 = a, y = b, z = c$. 
Unification

• **Unification**: matching literals and doing substitutions that resolution can be applied.

• **Substitution**: when a variable name is replaced by another variable or element of the domain.
  
  – Notation \([x/a]\) means replacing all occurrences of \(x\) with \(a\) in the formula
  
  – Example: substitution \([x/5]\) in \(p(x) \lor Q(x,y)\) results in \(p(5) \lor Q(5,y)\)
Unification

• It is an algorithm for determining the substitutions needed to make two predicate logic expressions match.

• A variable cannot be unified with a term containing that variable. The test for it is called the **occurs check**.
  – Example: cannot substitute $x$ for $x + y$ in $p(x + y)$
  – Most applicable when rather than having variables we have whole expressions (terms) evaluating to elements of the domain.
  – Example: $x + y$ is a term; when $x, y \in \mathbb{Z}$ and $x + y \in \mathbb{Z}$, with terms we can write formulas such as $p(x + y) \lor Q(y - 2)$
Algorithm to convert to clausal form (1)

(1) Eliminate conditionals \( \rightarrow \), using the equivalence
   \[ p \rightarrow q \equiv \neg p \lor q \]
   e.g. \((\exists x)(p(x) \land (\forall y)(f(y) \rightarrow h(x, y)))\) becomes
   \((\exists x)(p(x) \land (\forall y)(\neg f(y) \lor h(x, y)))\)

(2) Eliminate negations or reduce the scope of negation to one atom.
   e.g. \(\neg\neg p \equiv p\)
   \[\neg(p \land q) \equiv \neg p \lor \neg q\]
   \[\neg(\forall x \in S, F(x)) \equiv \exists x \in S, \neg F(x)\]
   \[\neg(\exists x \in S, F(x)) \equiv \forall x \in S, \neg F(x)\]

(3) Standardize variables within a well-formed formula so that the bound or free variables of each quantifier have unique names. e.g. \((\exists x)\neg p(x) \lor (\forall x)p(x)\) is replaced by \((\exists x)\neg p(x) \lor (\forall y)p(y)\)
Algorithm to convert to clausal form (2)

(4) Advanced step: if there are existential quantifiers, eliminate them by using Skolem functions
   e.g. $(\exists x)p(x)$ is replaced by $p(a)$
   $(\forall x)(\exists y)k(x, y)$ is replaced by $(\forall x)k(x, f(x))$

(5) Convert the formula to prenex form
   e.g. $(\exists x)(p(x) \land (\forall y)(\neg f(y) \lor h(x, y)))$ becomes
   $(\forall y)(p(a) \land (\neg f(y) \lor h(a, y)))$

(6) Convert the formulas to CNF, which is a conjunctive of clauses. Each clause is a disjunction.
   e.g. $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$

(7) Drop the universal quantifiers
   e.g. the formula in (5) becomes $p(a) \land (\neg f(y) \lor h(a, y))$
Algorithm to convert to clausal form (3)

(8) Eliminate the conjunctive signs by writing the formula as a set of clauses

e.g. \( p(a) \land (\neg f(y) \lor h(a, y)) \) becomes \( p(a), (\neg f(y) \lor h(a, y)) \)

(9) Rename variables in clauses, if necessary, so that the same variable name is only used in one clause.

e.g. \( p(x) \lor q(y) \lor k(x, y) \) and \( \neg p(x) \lor q(y) \) becomes
\( p(x) \lor q(y) \lor k(x, y) \) and \( \neg p(x) \lor q(y) \)
Example: Resolution for predicate logic

Anyone passing his history exams and winning the lottery is happy.

\( \forall X (\text{pass}(X,\text{history}) \land \text{win}(X,\text{lottery}) \rightarrow \text{happy}(X)) \)

Anyone who studies or is lucky can pass all his exams.

\( \forall X \forall Y (\text{study}(X) \lor \text{lucky}(X) \rightarrow \text{pass}(X,Y)) \)

John did not study but he is lucky.

\( \neg \text{study}(\text{john}) \land \text{lucky}(\text{john}) \)

Anyone who is lucky wins the lottery.

\( \forall X (\text{lucky}(X) \rightarrow \text{win}(X,\text{lottery})) \)

These four predicate statements are now changed to clause form (Section 12.2.2):

1. \( \neg \text{pass}(X,\text{history}) \lor \neg \text{win}(X,\text{lottery}) \lor \text{happy}(X) \)
2. \( \neg \text{study}(Y) \lor \text{pass}(Y,Z) \)
3. \( \neg \text{lucky}(W) \lor \text{pass}(W,V) \)
4. \( \neg \text{study}(\text{john}) \)
5. \( \text{lucky}(\text{john}) \)
6. \( \neg \text{lucky}(U) \lor \text{win}(U,\text{lottery}) \)

Into these clauses is entered, in clause form, the negation of the conclusion:

7. \( \neg \text{happy}(\text{john}) \)
\( \neg \text{pass}(X, \text{history}) \lor \neg \text{win}(X, \text{lottery}) \lor \text{happy}(X) \quad \neg \text{lucky}(U) \lor \text{win}(U, \text{lottery}) \)

\{U/X\}

\( \neg \text{pass}(U, \text{history}) \lor \text{happy}(U) \lor \neg \text{lucky}(U) \quad \neg \text{happy(\text{john})} \)

\{\text{\text{\text{\text{\text{john}/U}}}\}}\)

\( \text{\text{\text{\text{\text{{}\}}}}} \quad \neg \text{pass(\text{\text{\text{\text{\text{john}}/\text{history}}}\}}) \lor \neg \text{\text{\text{\text{\text{lucky(\text{\text{\text{\text{\text{john}}}})}\}}}}\}

\{\} \quad \neg \text{lucky(V) \lor \text{pass(V, W)}\}

\{\text{\text{\text{\text{\text{\text{john}/V}}} \text{\text{\text{\text{\text{history}/W}}}\}}\}

\( \neg \text{lucky(\text{\text{\text{\text{\text{john}}}})} \quad \text{lucky(\text{\text{\text{\text{\text{john}}}})} \}

\{\} \)