COL703: Logic for Computer Science (Jul-Nov 2023)

Lectures 23 & 24 (Predicate Resolution)

Kumar Madhukar

madhukar@cse.iitd.ac.in

November 02nd and 06th, 2023

Unification

- a substitution is a function θ from the set of σ-terms to itself such that $c\theta = c$ for each constant symbol c, and $f(t_1, \ldots, t_k)\theta = f(t_1\theta, \ldots, t_k\theta)$ for each k-ary function symbol f
- composition of substitutions is written diagrammatically ($\theta.\theta'$ denotes the substitution obtained by applying θ first, and then θ')
- given a set of literals $D = \{L_1, \ldots, L_k\}$ and a substitution θ, define $D\theta = \{L_1\theta, \ldots, L_k\theta\}$
- we say that θ unifies D if $D\theta = \{L\}$ for some literal L

Most General Unifier

- $\theta = [f(a)/x][a/y]$ unifies $\{P(x), P(f(y))\}$
- $\theta' = [f(y)/x]$ also unifies $\{P(x), P(f(y))\}$
- θ' is a more general unifier than θ (because $\theta = \theta'.[a/y]$)
- θ is a most general unifier of a set of literals D if θ is a unifier of D, and for any other unifier θ', we have that $\theta' = \theta.\theta''$
- most general unifiers are only unique up to renaming variables (why?)

Unification theorem

- a set of literals either has no unifier or it has a most general unifier
- $\{P(f(x)), P(g(x))\}$ cannot be unified
- $\{P(f(x)), P(x)\}$ cannot be unified
- we cannot unify a variable x and a term t is x occurs in t
- a unifiable set of literals has a most general unifier
- proof:

Robinson's algorithm

Unification Algorithm
Input: Set of literals D
Output: Either a most general unifier of D or "fail"

$\theta :=$ identity substitution
while θ is not a unifier of D **do**
begin
pick two distinct literals in $D\theta$ and find the left-most positions at which they differ

if one of the corresponding sub-terms is a variable x and the other a term t not containing x
then $\theta := \theta \cdot [t/x]$ **else** output "fail" and halt
end

Termination

- a variable x is replaced in each iteration with a term t that does not contain x
- the number of different variables occuring in $D\theta$ decreases by one in each iteration

Correctness

- for any unifier θ' of D, we have $\theta' = \theta.\theta'$
- argue that this is a loop invariant
- holds initially (θ is identity)
- why does the inductive step work?

Resolution

Definition 3 (Resolution)**.** Let C_1 and C_2 be clauses *with no variable in common.* We say that a clause R is a *resolvent* of C_1 and C_2 if there are sets of literals $D_1 \subseteq C_1$ and $D_2 \subseteq C_2$ such that $D_1 \cup \overline{D_2}$ has a most general unifier θ, and

$$R = (C_1\theta \setminus \{L\}) \cup (C_2\theta \setminus \{\overline{L}\}), \tag{1}$$

where $L = D_1\theta$ and $\overline{L} = D_2\theta$. More generally, if C_1 and C_2 are arbitrary clauses, we say that R is a resolvent of C_1 and C_2 if there are variable renamings θ_1 and θ_2 such that $C_1\theta_1$ and $C_2\theta_2$ have no variable in common, and R is a resolvent of $C_1\theta_1$ and $C_2\theta_2$ according to the definition above.

Example

$\{P(f(x), g(y)), Q(x, y)\}$

$\{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$

Example

$\{P(f(x), g(y)), Q(x, y)\}$ $\{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$

check if there are common variables

Example

$\{P(f(x), g(y)), Q(x, y)\}$ $\{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$

check if there are common variables

pick D_1 and D_2, and get a most general unifier θ of $D_1 \cup \overline{D_2}$

Example

$\{P(f(x), g(y)), Q(x, y)\}$ $\{\neg P(f(f(a)), g(z)), Q(f(a), z)\}$

check if there are common variables

pick D_1 and D_2, and get a most general unifier θ of $D_1 \cup \overline{D_2}$

resolve, to get $\{Q(f(a), z)\}$

Another example

$\{P(x), P(y)\}$

$\{\neg P(x), \neg P(y)\}$

Resolution procedure

Input: a set of clauses, S

Output: If the algorithm terminates, report that S is sat or unsat

$S_0 := S$

Choose clashing clauses $C_1, C_2 \in S_i$, and let $C = Res(C_1, C_2)$.

If C is $\square$, terminate and report unsat

$S_{i+1} = S_i \cup C$

If $S_{i+1} = S_i$ for all possible pairs of clashing clauses, terminate and report sat

Resolution procedure

Input: a set of clauses, S

Output: If the algorithm terminates, report that S is sat or unsat

$S_0 := S$

Choose clashing clauses $C_1, C_2 \in S_i$, and let $C = Res(C_1, C_2)$.

If C is $\square$, terminate and report unsat

$S_{i+1} = S_i \cup C$

If $S_{i+1} = S_i$ for all possible pairs of clashing clauses, terminate and report sat

this may not terminate for a satisfiable set of clauses (because of existence of infinite models); so this is not a decision procedure

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6
9. $\{Q(a), R'(f(a))\}$ [a/x] 2,4

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6
9. $\{Q(a), R'(f(a))\}$ [a/x] 2,4
10. $\{R'(f(a))\}$ 8,9

Example

1.	$\{\neg P(x), Q(x), R(x, f(x))\}$	given
2.	$\{\neg P(x), Q(x), R'(f(x))\}$	given
3.	$\{P'(a)\}$	given
4.	$\{P(a)\}$	given
5.	$\{\neg R(a, y), P'(y)\}$	given
6.	$\{\neg P'(x), \neg Q(x)\}$	given
7.	$\{\neg P'(x), \neg R'(x)\}$	given
8.	$\{\neg Q(a)\}$	[a/x] 3,6
9.	$\{Q(a), R'(f(a))\}$	[a/x] 2,4
10.	$\{R'(f(a))\}$	8,9
11.	$\{Q(a), R(a, f(a))\}$	[a/x] 1,4

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6
9. $\{Q(a), R'(f(a))\}$ [a/x] 2,4
10. $\{R'(f(a))\}$ 8,9
11. $\{Q(a), R(a, f(a))\}$ [a/x] 1,4
12. $\{R(a, f(a))\}$ 8,11

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6
9. $\{Q(a), R'(f(a))\}$ [a/x] 2,4
10. $\{R'(f(a))\}$ 8,9
11. $\{Q(a), R(a, f(a))\}$ [a/x] 1,4
12. $\{R(a, f(a))\}$ 8,11
13. $\{P'(f(a))\}$ [f(a)/y] 5,12

Example

1. $\{\neg P(x), Q(x), R(x, f(x))\}$ given
2. $\{\neg P(x), Q(x), R'(f(x))\}$ given
3. $\{P'(a)\}$ given
4. $\{P(a)\}$ given
5. $\{\neg R(a, y), P'(y)\}$ given
6. $\{\neg P'(x), \neg Q(x)\}$ given
7. $\{\neg P'(x), \neg R'(x)\}$ given

8. $\{\neg Q(a)\}$ [a/x] 3,6
9. $\{Q(a), R'(f(a))\}$ [a/x] 2,4
10. $\{R'(f(a))\}$ 8,9
11. $\{Q(a), R(a, f(a))\}$ [a/x] 1,4
12. $\{R(a, f(a))\}$ 8,11
13. $\{P'(f(a))\}$ [f(a)/y] 5,12
14. $\{\neg R'(f(a))\}$ [f(a)/x] 7,13

Example

1.	$\{\neg P(x), Q(x), R(x, f(x))\}$	given
2.	$\{\neg P(x), Q(x), R'(f(x))\}$	given
3.	$\{P'(a)\}$	given
4.	$\{P(a)\}$	given
5.	$\{\neg R(a, y), P'(y)\}$	given
6.	$\{\neg P'(x), \neg Q(x)\}$	given
7.	$\{\neg P'(x), \neg R'(x)\}$	given
8.	$\{\neg Q(a)\}$	[a/x] 3,6
9.	$\{Q(a), R'(f(a))\}$	[a/x] 2,4
10.	$\{R'(f(a))\}$	8,9
11.	$\{Q(a), R(a, f(a))\}$	[a/x] 1,4
12.	$\{R(a, f(a))\}$	8,11
13.	$\{P'(f(a))\}$	[f(a)/y] 5,12
14.	$\{\neg R'(f(a))\}$	[f(a)/x] 7,13
15.	$\{\}$	10,14

Another example

1. $\{\neg P(x, y), P(y, x)\}$ given
2. $\{\neg P(x, y), \neg P(y, z), P(x, z)\}$ given
3. $\{P(x, f(x))\}$ given
4. $\{\neg P(x, x)\}$ given

Exercise

Consider the following sentences over a signature containing a ternary predicate symbol A, a constant symbol e, and a unary function symbol s.

$F_1 : \forall x \; A(e, x, x)$

$F_2 : \forall x \forall y \forall z \; (\neg A(x, y, z) \vee A(s(x), y, s(z)))$

$F_3 : \forall x \exists y \; A(s(s(e)), x, y)$

Use first-order resolution to show that $F_1 \wedge F_2 \models F_3$.

Exercise

Consider the following sentences over a signature containing a ternary predicate symbol A, a constant symbol e, and a unary function symbol s.

$F_1 : \forall x \; A(e, x, x)$

$F_2 : \forall x \forall y \forall z \; (\neg A(x, y, z) \vee A(s(x), y, s(z)))$

$F_3 : \forall x \exists y \; A(s(s(e)), x, y)$

Use first-order resolution to show that $F_1 \wedge F_2 \models F_3$.

In other words, show that $F_1 \wedge F_2 \wedge \neg F_3$ is unsatisfiable.

Resolution Lemma

- Given a formula H with free variables $x_1, \ldots, x_n$, its universal closure $\forall^* \; H$ is the sentence $\forall x_1, \ldots, \forall x_n \; H$.

- Let $F = \forall x_1, \ldots, \forall x_n \; G$ be a closed formula in Skolem form, with G quantifier-free. Let R be a resolvent of two clauses in G. Then $F \equiv \forall^* \; (G \cup \{R\})$.

- Soundness follows immediately from this.

Lifting Lemma

Let C_1 and C_2 be clauses with respective ground instances G_1 and G_2. Suppose that R is a propositional resolvent of G_1 and G_2. Then C_1 and C_2 have a predicate-logic resolvent R' such that R is a ground instance of R'.

Proof:

Reference material: `https://www.cs.ox.ac.uk/people/james.worrell/lecture14-2015.pdf`

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\Box$ from F'.

Proof:

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\Box$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \Box$

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\Box$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \Box$
- C'_i is either a ground instance of a clause in F' or is a resolvent of C'_j and C'_k for $j, k < i$

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\Box$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \Box$
- C'_i is either a ground instance of a clause in F' or is a resolvent of C'_j and C'_k for $j, k < i$
- we inductively define a corresponding predicate-logic proof $C'_1, C'_2, \ldots, C_n = \Box$ such that C'_i is a ground instance of C_i

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\square$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \square$
- C'_i is either a ground instance of a clause in F' or is a resolvent of C'_j and C'_k for $j, k < i$
- we inductively define a corresponding predicate-logic proof $C'_1, C'_2, \ldots, C_n = \square$ such that C'_i is a ground instance of C_i
- if C'_i is a ground instance of $C \in F'$, $C_i = C$

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\square$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \square$
- C'_i is either a ground instance of a clause in F' or is a resolvent of C'_j and C'_k for $j, k < i$
- we inductively define a corresponding predicate-logic proof $C'_1, C'_2, \ldots, C_n = \square$ such that C'_i is a ground instance of C_i
- if C'_i is a ground instance of $C \in F'$, $C_i = C$
- otherwise, C'_i is a resolvent of C'_j and C'_k for $j, k < i$

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\square$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C_1', C_2', \ldots, C_n' = \square$
- C_i' is either a ground instance of a clause in F' or is a resolvent of C_j' and C_k' for $j, k < i$
- we inductively define a corresponding predicate-logic proof $C_1', C_2', \ldots, C_n = \square$ such that C_i' is a ground instance of C_i
- if C_i' is a ground instance of $C \in F'$, $C_i = C$
- otherwise, C_i' is a resolvent of C_j' and C_k' for $j, k < i$
- by induction, we have constructed C_j and C_k ...

Refutation Completeness

Let F be a closed formula in Skolem form with its matrix F' in CNF. If F is unsat, then there is a predicate-logic resolution proof of $\Box$ from F'.

Proof:

- by completeness of ground resolution, there is a proof $C'_1, C'_2, \ldots, C'_n = \Box$
- C'_i is either a ground instance of a clause in F' or is a resolvent of C'_j and C'_k for $j, k < i$
- we inductively define a corresponding predicate-logic proof $C'_1, C'_2, \ldots, C_n = \Box$ such that C'_i is a ground instance of C_i
- if C'_i is a ground instance of $C \in F'$, $C_i = C$
- otherwise, C'_i is a resolvent of C'_j and C'_k for $j, k < i$
- by induction, we have constructed C_j and C_k ...
- by the lifting lemma ...

Thank you!