Logics and Statistics for Language Modeling

Carlos Areces

areces@loria.fr
http://www.loria.fr/~areces/ls
INRIA Nancy Grand Est
Nancy, France

2009/2010

Today's Program

Today's Program

- Resolution for FOL
  - Unification
  - Clausal Form. Skolemization.
  - Unification
  - The Resolution Rules
  - Non Termination

Conventions and Notation

Conventions and Notation

- x, y, z denote variables.
- a, b, c denote constants.
- f , g, h denote function.
- s, t, r denote arbitrary terms.

Conventions and Notation

- x, y, z denote variables.
- a, b, c denote constants.
- f , g, h denote function.
- s, t, r denote arbitrary terms.

Examples:

- $f(x, g(x, a), y)$ is a term, where f is ternary, g is binary, a is constant.

Substitutions

Substitutions

Substitution

- A mapping from variables to terms, where all but finitely many variables are mapped to themselves.

Substitutions

Substitution

- A mapping from variables to terms, where all but finitely many variables are mapped to themselves.

Example

A substitution is represented as a set of bindings:

- $\{x \mapsto f(a,b), y \mapsto z\}$.
- $\{x \mapsto f(x,y), y \mapsto f(x,y)\}$.

All variables except x and y are mapped to themselves by these substitutions.

Substitution Application

Substitution Application

Applying a substitution σ to a term t:

$$t\sigma = \begin{cases} \sigma(x) & \text{if } t = x \\ f(t_1\sigma, \ldots, t_n\sigma) & \text{if } t = f(t_1, ..., t_n) \end{cases}$$

Substitution Application

Applying a substitution σ to a term t:

$$t\sigma = \begin{cases} \sigma(x) & \text{if } t = x \\ f(t_1\sigma, \ldots, t_n\sigma) & \text{if } t = f(t_1, ..., t_n) \end{cases}$$

Example

- $\sigma = \{x \mapsto f(x, y), y \mapsto g(a)\}$.
- $t = f(x, g(f(x, f(y, z))))$.
- $t\sigma = f(f(x, y), g(f(f(x, y), f(g(a), z))))$.

Substitution Application

Applying a substitution σ to a term t:

$$t\sigma = \begin{cases} \sigma(x) & \text{if } t = x \\ f(t_1\sigma, \ldots, t_n\sigma) & \text{if } t = f(t_1, ..., t_n) \end{cases}$$

Example

- $\sigma = \{x \mapsto f(x, y), y \mapsto g(a)\}$.
- $t = f(x, g(f(x, f(y, z))))$.
- $t\sigma = f(f(x, y), g(f(f(x, y), f(g(a), z))))$.

A substitution σ is a unifier of the terms s and t if $s\sigma = t\sigma$.

Unifier, Most General Unifier

Unifier, Most General Unifier

Unification Problem: $f(x, z) \doteq^? f(y, g(a))$.

Unifier, Most General Unifier

Unification Problem: $f(x, z) \doteq^? f(y, g(a))$.

- Some of the unifiers:

$$\{x \mapsto y, z \mapsto g(a)\}$$
$$\{y \mapsto x, z \mapsto g(a)\}$$
$$\{x \mapsto a, y \mapsto a, z \mapsto g(a)\}$$
$$\{x \mapsto g(a), y \mapsto g(a), z \mapsto g(a)\}$$
$$\{x \mapsto f(x, y), y \mapsto f(x, y), z \mapsto g(a)\}$$
$$\ldots$$

Unifier, Most General Unifier

Unification Problem: $f(x, z) \doteq^? f(y, g(a))$.

- Some of the unifiers:

$$\{x \mapsto y, z \mapsto g(a)\}$$
$$\{y \mapsto x, z \mapsto g(a)\}$$
$$\{x \mapsto a, y \mapsto a, z \mapsto g(a)\}$$
$$\{x \mapsto g(a), y \mapsto g(a), z \mapsto g(a)\}$$
$$\{x \mapsto f(x, y), y \mapsto f(x, y), z \mapsto g(a)\}$$
$$\ldots$$

Most General Unifiers (mgu):

$$\{x \mapsto y, z \mapsto g(a)\}, \{y \mapsto x, z \mapsto g(a)\}.$$

- mgu is unique up to a variable renaming.

Unification Algorithm

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^{?} t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^? t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Naive Algorithm: Write down two terms and set markers at the beginning of the terms. Then:

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^? t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Naive Algorithm: Write down two terms and set markers at the beginning of the terms. Then:

1. Move markers simultaneously, one symbol at a time, until both move off the end of the term (success), or until they point to two different symbols;

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^? t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Naive Algorithm: Write down two terms and set markers at the beginning of the terms. Then:

1. Move markers simultaneously, one symbol at a time, until both move off the end of the term (success), or until they point to two different symbols;
2. If the two symbols are both non-variables, then fail; otherwise, one is a variable (call it x) and the other one is the first symbol of a subterm (call it t):

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^? t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Naive Algorithm: Write down two terms and set markers at the beginning of the terms. Then:

1. Move markers simultaneously, one symbol at a time, until both move off the end of the term (success), or until they point to two different symbols;
2. If the two symbols are both non-variables, then fail; otherwise, one is a variable (call it x) and the other one is the first symbol of a subterm (call it t):
   2.1 If x occurs in t, then fail;

Unification Algorithm

Goal: Design algorithm that for a given problem $s \doteq^? t$

- returns an mgu of s and t if they are unifiable,
- reports failure otherwise.

Naive Algorithm: Write down two terms and set markers at the beginning of the terms. Then:

1. Move markers simultaneously, one symbol at a time, until both move off the end of the term (success), or until they point to two different symbols;
2. If the two symbols are both non-variables, then fail; otherwise, one is a variable (call it x) and the other one is the first symbol of a subterm (call it t):
   2.1 If x occurs in t, then fail;
   2.2 Else, replace x everywhere by t (including in the solution), print "$x \mapsto t$" as a partial solution. Go to 1.

Naive Algorithm

Naive Algorithm

- Finds disagreements in the two terms to be unified.
- Attempts to repair the disagreements by binding variables to terms.
- Fails when function symbols clash, or when an attempt is made to unify a variable with a term containing that variable.

Example

$f(x, g(a), g(z))$

$f(g(y), g(y), g(g(x)))$

Naive Algorithm

- Finds disagreements in the two terms to be unified.
- Attempts to repair the disagreements by binding variables to terms.
- Fails when function symbols clash, or when an attempt is made to unify a variable with a term containing that variable.

Example

$f(x, g(a), g(z))$

$f(g(y), g(y), g(g(x)))$

We can also unify formulas, we just consider them as if they were terms.

Resolution for Propositional Logic

Resolution for Propositional Logic

- Clausal Form. Write φ in conjunctive normal form
  $\varphi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$,

Resolution for Propositional Logic

- Clausal Form. Write φ in conjunctive normal form
  $\varphi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$,
  and let the clause set associated to φ be

$$ClSet(\varphi) = \{\{\psi_{l,m} \mid m \in M\} \mid l \in L\}.$$

Resolution for Propositional Logic

- Clausal Form. Write φ in conjunctive normal form
  $\varphi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$,
  and let the clause set associated to φ be

$$ClSet(\varphi) = \{\{\psi_{l,m} \mid m \in M\} \mid l \in L\}.$$

- Let $ClSet^*(\varphi)$ be the smallest set containing $ClSet(\varphi)$ and closed under the (RES) rule:

$$\frac{Cl_1 \cup \{N\} \in ClSet^*(\varphi) \qquad Cl_2 \cup \{\neg N\} \in ClSet^*(\varphi)}{Cl_1 \cup Cl_2 \in ClSet^*(\varphi)}$$

- I.e., we apply (RES) to the set of clauses till we cannot obtain any new clause. We obtain the empty clause $\{\}$, if and only if the original formula was UNSAT.

Resolution for Propositional Logic

Resolution for Propositional Logic

1. $(p \wedge q) \wedge (p \rightarrow r) \wedge (r \rightarrow t) \wedge (t \rightarrow \neg q)$

Resolution for Propositional Logic

1. $(p \wedge q) \wedge (p \rightarrow r) \wedge (r \rightarrow t) \wedge (t \rightarrow \neg q)$
2. $(p \wedge q) \wedge (\neg p \vee r) \wedge (\neg r \vee t) \wedge (\neg t \vee \neg q)$

Resolution for Propositional Logic

1. $(p \wedge q) \wedge (p \rightarrow r) \wedge (r \rightarrow t) \wedge (t \rightarrow \neg q)$
2. $(p \wedge q) \wedge (\neg p \vee r) \wedge (\neg r \vee t) \wedge (\neg t \vee \neg q)$
3. $\{\{p\}, \{q\}\{\neg p, r\}, \{\neg r, t\}, \{\neg t, \neg q\}\}$

Resolution for Propositional Logic

1. $(p \wedge q) \wedge (p \rightarrow r) \wedge (r \rightarrow t) \wedge (t \rightarrow \neg q)$
2. $(p \wedge q) \wedge (\neg p \vee r) \wedge (\neg r \vee t) \wedge (\neg t \vee \neg q)$
3. $\{\{p\}, \{q\}\{\neg p, r\}, \{\neg r, t\}, \{\neg t, \neg q\}\}$
4. $\{\{p\}, \{q\}\{\neg p, r\}, \{\neg r, t\}, \{\neg t, \neg q\}, \{r\}, \{t\}, \{\neg q\}, \{\}\}$

Can we do the “same” for FO?

Can we do the “same” for FO?

There are two problems

Can we do the "same" for FO?

There are two problems

- What do we do with quantifiers?

Can we do the “same” for FO?

There are two problems

- What do we do with quantifiers?

Eliminate them $\Rightarrow$ Skolemnization

Can we do the "same" for FO?

There are two problems

- What do we do with quantifiers?

  Eliminate them $\Rightarrow$ Skolemnization

- The (RES) formula is slightly too weak

Can we do the "same" for FO?

There are two problems

- What do we do with quantifiers?

  Eliminate them $\Rightarrow$ Skolemnization

- The (RES) formula is slightly too weak

$\{\{\forall x.P(x)\},\{\neg P(a)\}\}\}$ is inconsistent
but $\{\}$ cannot be derived by (RES) as it stand for PL
$\Rightarrow$
Unification

Some Properties of Quantifiers

Some Properties of Quantifiers

- $\forall x.\forall y.\varphi$ is the same as $\forall y.\forall x.\varphi$
- $\exists x.\exists y.\varphi$ is the same as $\exists y.\exists x.\varphi$
- $\exists x.\forall y.\varphi$ is not the same as $\forall y.\exists x.\varphi$
- $\forall x.\varphi$ is the same as $\forall y.\varphi[x/y]$ if y does not appear in φ, and similarly for $\exists x.\varphi$ and $\exists y.\varphi[x/y]$.
- $\varphi \wedge Qx.\psi$ is the same as $Qx.(\varphi \wedge \psi)$ if x does not appear in φ ($Q \in \{\forall, \exists\}$).
- $\neg\exists x.\varphi$ is equivalent to $\forall x.\neg\varphi$ and $\neg\forall y.\varphi$ is equivalent to $\exists x.\neg\varphi$.

Clausal Form and Skolemnization

Clausal Form and Skolemnization

- Write φ in prenex normal form (PNF), with the matrix in conjunctive normal form:

$$\varphi = \mathcal{Q}.\psi \text{ where } \psi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$$

Clausal Form and Skolemnization

- Write φ in prenex normal form (PNF), with the matrix in conjunctive normal form:

$$\varphi = \mathcal{Q}.\psi \text{ where } \psi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$$

- Let $Sko(\varphi)$ be the "skolemnization" of φ, obtained as follows

Clausal Form and Skolemnization

- Write φ in prenex normal form (PNF), with the matrix in conjunctive normal form:

$$\varphi = \mathcal{Q}.\psi \text{ where } \psi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$$

- Let $Sko(\varphi)$ be the "skolemnization" of φ, obtained as follows
  - While there is an existential quantifier in $\mathcal{Q}$, let $\bar{x}$ be the list of variables universally quantified in $\mathcal{Q}$ which occur in front of the first existential quantifier $\exists x_i$.

Clausal Form and Skolemnization

- Write φ in prenex normal form (PNF), with the matrix in conjunctive normal form:

$$\varphi = \mathcal{Q}.\psi \text{ where } \psi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$$

- Let $Sko(\varphi)$ be the "skolemnization" of φ, obtained as follows
  - While there is an existential quantifier in $\mathcal{Q}$, let $\bar{x}$ be the list of variables universally quantified in $\mathcal{Q}$ which occur in front of the first existential quantifier $\exists x_i$.
  - Eliminate $\exists x_i$ from $\mathcal{Q}$ and replace ψ by $\psi[f(\bar{x})/x_i]$ where f is a fresh $|\bar{x}|$-ary function not used before.

Clausal Form and Skolemnization

- Write φ in prenex normal form (PNF), with the matrix in conjunctive normal form:

$$\varphi = \mathcal{Q}.\psi \text{ where } \psi = \bigwedge_{l \in L} \bigvee_{m \in M} \psi_{(l,m)}$$

- Let $Sko(\varphi)$ be the "skolemnization" of φ, obtained as follows
  - While there is an existential quantifier in $\mathcal{Q}$, let $\bar{x}$ be the list of variables universally quantified in $\mathcal{Q}$ which occur in front of the first existential quantifier $\exists x_i$.
  - Eliminate $\exists x_i$ from $\mathcal{Q}$ and replace ψ by $\psi[f(\bar{x})/x_i]$ where f is a fresh $|\bar{x}|$-ary function not used before.
- After eliminating all the existential quantifiers, drop $\mathcal{Q}$, consider the obtained matrix as a propositional formula in conjunctive normal form and define $ClSet$ as we did before.

Skolemnization Examples

Skolemnization Examples

1 $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$

Skolemnization Examples

1. $\exists x.\forall y.\exists z.(P(x,y) \land P(y,z) \to P(x,z))$
   Sk: $P(c,y) \land P(y,f(y)) \to P(c,f(y))$

Skolemnization Examples

1 $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$
Sk: $P(c,y) \wedge P(y,f(y)) \rightarrow P(c,f(y))$

2 $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$

Skolemnization Examples

1. $\exists x.\forall y.\exists z.(P(x,y) \land P(y,z) \rightarrow P(x,z))$
   Sk: $P(c,y) \land P(y,f(y)) \rightarrow P(c,f(y))$

2. $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$
   Sk: $P(x,y) \rightarrow (P(x,f(x,y)) \rightarrow P(f(x,y),y))$

Skolemnization Examples

1 $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$
Sk: $P(c,y) \wedge P(y,f(y)) \rightarrow P(c,f(y))$

2 $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$
Sk: $P(x,y) \rightarrow (P(x,f(x,y)) \rightarrow P(f(x,y),y))$

3 $\forall x.\exists y.P(y,y)$

Skolemnization Examples

1. $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$
   Sk: $P(c,y) \wedge P(y,f(y)) \rightarrow P(c,f(y))$

2. $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$
   Sk: $P(x,y) \rightarrow (P(x,f(x,y)) \rightarrow P(f(x,y),y))$

3. $\forall x.\exists y.P(y,y)$
   Sk: $P(f(x),f(x))$

Skolemnization Examples

1. $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$
   Sk: $P(c,y) \wedge P(y,f(y)) \rightarrow P(c,f(y))$

2. $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$
   Sk: $P(x,y) \rightarrow (P(x,f(x,y)) \rightarrow P(f(x,y),y))$

3. $\forall x.\exists y.P(y,y)$
   Sk: $P(f(x),f(x))$

4. $\exists x.\forall x.P(x,x)$

Skolemnization Examples

1. $\exists x.\forall y.\exists z.(P(x,y) \wedge P(y,z) \rightarrow P(x,z))$
   Sk: $P(c,y) \wedge P(y,f(y)) \rightarrow P(c,f(y))$

2. $\forall x.\forall y.(P(x,y) \rightarrow \exists z.(P(x,z) \rightarrow P(z,y)))$
   Sk: $P(x,y) \rightarrow (P(x,f(x,y)) \rightarrow P(f(x,y),y))$

3. $\forall x.\exists y.P(y,y)$
   Sk: $P(f(x),f(x))$

4. $\exists x.\forall x.P(x,x)$
   Sk: $P(x,x)$

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$ Not Unifiable

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$ Not Unifiable

2. $f(x, f(y))$ with $f(a, z)$

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$ Not Unifiable
2. $f(x, f(y))$ with $f(a, z)$ Unifiable

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$ Not Unifiable
2. $f(x, f(y))$ with $f(a, z)$ Unifiable
3. $f(x, f(x))$ with $f(y, y)$

Examples of Unification

1.	$f(x, f(x))$ with $g(x, f(x))$	Not Unifiable
2.	$f(x, f(y))$ with $f(a, z)$	Unifiable
3.	$f(x, f(x))$ with $f(y, y)$	Not Unifiable

Examples of Unification

1. $f(x, f(x))$ with $g(x, f(x))$ — Not Unifiable
2. $f(x, f(y))$ with $f(a, z)$ — Unifiable
3. $f(x, f(x))$ with $f(y, y)$ — Not Unifiable
4. $f(x, f(x))$ with $f(g(y), f(g(y)))$

Examples of Unification

1.	$f(x, f(x))$ with $g(x, f(x))$	Not Unifiable
2.	$f(x, f(y))$ with $f(a, z)$	Unifiable
3.	$f(x, f(x))$ with $f(y, y)$	Not Unifiable
4.	$f(x, f(x))$ with $f(g(y), f(g(y)))$	Unifiable

Resolution for First Order Logic

Resolution for First Order Logic

- Let $ClSet^*(\varphi)$ be the smallest set containing $ClSet(\varphi)$ and clause under the (RES) and (FAC) rules:

$$[RES]\ \frac{Cl_1 \cup \{N\} \in ClSet^*(\varphi) \qquad Cl_2 \cup \{\neg M\} \in ClSet^*(\varphi)}{(Cl_1 \cup Cl_2)\theta \in ClSet^*(\varphi)}$$

$$[FAC]\ \frac{Cl \cup \{N, M\} \in ClSet^*(\varphi)}{(Cl \cup \{N\})\theta \in ClSet^*(\varphi)}$$

where θ is the most general unifier of M and N.

Resolution for First Order Logic

- Let $ClSet^*(\varphi)$ be the smallest set containing $ClSet(\varphi)$ and clause under the (RES) and (FAC) rules:

$$[RES]\ \frac{Cl_1 \cup \{N\} \in ClSet^*(\varphi) \qquad Cl_2 \cup \{\neg M\} \in ClSet^*(\varphi)}{(Cl_1 \cup Cl_2)\theta \in ClSet^*(\varphi)}$$

$$[FAC]\ \frac{Cl \cup \{N, M\} \in ClSet^*(\varphi)}{(Cl \cup \{N\})\theta \in ClSet^*(\varphi)}$$

  where θ is the most general unifier of M and N.

- **Important:** Before applying the [RES] rule, rename variables in the clauses so that they don't share any variable.
- **Theorem:** $\backslash\forall\varphi$, $ClSet^*\ \varphi$ is inconsistent iff $\{\} \in ClSet^*(\varphi)$.

Example

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)
4. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall y(\neg Q(y))) \wedge \exists z(P(z)))$ (move to PNF)

Example

1. $\neg((\forall x(P(x) \to Q(x)) \wedge \forall x(\neg Q(x))) \to \forall x(\neg P(x)))$ (eliminate $\to$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)
4. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall y(\neg Q(y))) \wedge \exists z(P(z)))$ (move to PNF)
5. $\exists z \forall y \forall x(((\neg P(x) \vee Q(x)) \wedge (\neg Q(y))) \wedge (P(z)))$ (skolemize)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)
4. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall y(\neg Q(y))) \wedge \exists z(P(z)))$ (move to PNF)
5. $\exists z \forall y \forall x(((\neg P(x) \vee Q(x)) \wedge (\neg Q(y))) \wedge (P(z)))$ (skolemize)
6. $(((\neg P(x) \vee Q(x)) \wedge \neg Q(y)) \wedge P(c))$ (write as clasues)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)
4. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall y(\neg Q(y))) \wedge \exists z(P(z)))$ (move to PNF)
5. $\exists z \forall y \forall x(((\neg P(x) \vee Q(x)) \wedge (\neg Q(y))) \wedge (P(z)))$ (skolemize)
6. $(((\neg P(x) \vee Q(x)) \wedge \neg Q(y)) \wedge P(c))$ (write as clasues)
7. $\{\{\neg P(x), Q(x)\}, \{\neg Q(y)\}, \{P(c)\}\}$ (resolve)

Example

1. $\neg((\forall x(P(x) \rightarrow Q(x)) \wedge \forall x(\neg Q(x))) \rightarrow \forall x(\neg P(x)))$ (eliminate $\rightarrow$)
2. $\neg(\neg(\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \vee \forall x(\neg P(x)))$ (push $\neg$ in)
3. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall x(\neg Q(x))) \wedge \exists x(P(x)))$ (rename)
4. $((\forall x(\neg P(x) \vee Q(x)) \wedge \forall y(\neg Q(y))) \wedge \exists z(P(z)))$ (move to PNF)
5. $\exists z \forall y \forall x(((\neg P(x) \vee Q(x)) \wedge (\neg Q(y))) \wedge (P(z)))$ (skolemize)
6. $(((\neg P(x) \vee Q(x)) \wedge \neg Q(y)) \wedge P(c))$ (write as clasues)
7. $\{\{\neg P(x), Q(x)\}, \{\neg Q(y)\}, \{P(c)\}\}$ (resolve)
8. $\{\{\neg P(x), Q(x)\}, \{\neg Q(y)\}, \{P(c)\}, \{\neg P(z)\}, \{Q(c)\}, \{\}\}$ (UNSAT)

Termination?

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$
2. $\{\neg R(c,w),\ \neg P(w),\ P(f(w))\}$

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$
2. $\{\neg R(c,w),\ \neg P(w),\ P(f(w))\}$

With one resolution step we obtained

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \land P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$
2. $\{\neg R(c,w),\ \neg P(w),\ P(f(w))\}$

With one resolution step we obtained

3. $\{\neg R(c,c),\ \neg P(c),\ \neg P(f(c)),\ P(f^2(c))\}$

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$
2. $\{\neg R(c,w),\ \neg P(w),\ P(f(w))\}$

With one resolution step we obtained

3. $\{\neg R(c,c),\ \neg P(c),\ \neg P(f(c)),\ P(f^2(c))\}$
4. $\{\neg R(c,f(w)),\ R(f(w),f^2(w)),\ \neg R(c,w),\ \neg P(w)\}$.

Termination?

Let's consider the formula

$$\exists x \forall y (R(x,y) \rightarrow (P(y) \rightarrow \exists z.(R(y,z) \wedge P(z))))$$

1. $\{\neg R(c,y),\ \neg P(y),\ R(y,f(y))\}$
2. $\{\neg R(c,w),\ \neg P(w),\ P(f(w))\}$

With one resolution step we obtained

3. $\{\neg R(c,c),\ \neg P(c),\ \neg P(f(c)),\ P(f^2(c))\}$
4. $\{\neg R(c,f(w)),\ R(f(w),f^2(w)),\ \neg R(c,w),\ \neg P(w)\}$.

Clauses 2 y 4 resolve to give

5. $\{\neg R(c,f^2(w)),\ R(f^2(w),f^3(w)),\ \neg R(c,f(w)),\ \neg R(c,w),$ $\neg P(w)\}$.

State Explosion

State Explosion

- As the example we just saw shows, generating new clauses is easy.

State Explosion

- As the example we just saw shows, generating new clauses is easy.
- Indeed, if we are not careful we can easily generate millions of clauses leading nowhere.

State Explosion

- As the example we just saw shows, generating new clauses is easy.
- Indeed, if we are not careful we can easily generate millions of clauses leading nowhere.
- Notice that every time we generate a formula implied by another formula already in the clause set we are wasting time.

State Explosion

- As the example we just saw shows, generating new clauses is easy.
- Indeed, if we are not careful we can easily generate millions of clauses leading nowhere.
- Notice that every time we generate a formula implied by another formula already in the clause set we are wasting time.
- Discovering when this is happening to be able to avoid it, is where most FO provers spend their computing time (simplification and subsumption)

State Explosion

- As the example we just saw shows, generating new clauses is easy.
- Indeed, if we are not careful we can easily generate millions of clauses leading nowhere.
- Notice that every time we generate a formula implied by another formula already in the clause set we are wasting time.
- Discovering when this is happening to be able to avoid it, is where most FO provers spend their computing time (simplification and subsumption)
- The "no redundancy" constraint helps us keep the clause set under control, as we will reach sooner the point of saturation, where no new, non redundant clauses can can be generated.

Exercises

- Apply the resolution method to the following formula, to determine whether it's satisfiable:

$$\forall x . \exists y . (R(x, y) \rightarrow Q(y)) \wedge \forall y . \neg Q(y)$$

- Now try with

$$\forall x . \exists y . (R(x, y) \rightarrow Q(y)) \wedge \forall y . \neg Q(y) \wedge \exists x . R(x, x)$$