Automated Reasoning

Week 12. Predicate and Equational Logic

Cynthia Kop

Fall 2024

Recap

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

- **Specify** the goal in predicate logic.

$$(\exists x[\mathtt{S}(x) \wedge \forall y[\mathtt{L}(y) \rightarrow \mathtt{A}(x,y)]]) \wedge$$
$$(\forall x[(\mathtt{L}(x) \wedge \mathtt{B}(x)) \rightarrow \neg\exists y[\mathtt{S}(y) \wedge \mathtt{A}(y,x)]]) \;\rightarrow$$
$$\neg\exists x[\mathtt{L}(x) \wedge \mathtt{B}(x)]$$

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

- **Specify** the goal in predicate logic.
- **Negate** it.

$$(\exists x[\mathtt{S}(x) \wedge \forall y[\mathtt{L}(y) \rightarrow \mathtt{A}(x,y)]]) \wedge$$
$$(\forall x[(\mathtt{L}(x) \wedge \mathtt{B}(x)) \rightarrow \neg\exists y[\mathtt{S}(y) \wedge \mathtt{A}(y,x)]]) \wedge$$
$$\exists x[\mathtt{L}(x) \wedge \mathtt{B}(x)]$$

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

- **Specify** the goal in predicate logic.
- **Negate** it.
- Transform the formula into **Prenex normal form**

$$\exists x \exists v \forall y \forall z \forall u [\mathtt{S}(x) \wedge (\neg \mathtt{L}(y) \vee \mathtt{A}(x,y)) \wedge$$
$$(\neg \mathtt{L}(z) \vee \neg \mathtt{B}(z)) \vee (\neg \mathtt{S}(u) \vee \neg \mathtt{A}(u,z)) \wedge$$
$$\mathtt{L}(v) \wedge \mathtt{B}(v)]$$

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

- **Specify** the goal in predicate logic.
- **Negate** it.
- Transform the formula into **Prenex normal form**
- Use **Skolemization** to replace existential variables by function symbols

$$\forall y \forall z \forall u [\mathtt{S}(\mathtt{a}) \land (\neg \mathtt{L}(y) \lor \mathtt{A}(\mathtt{a}, y)) \land$$
$$(\neg \mathtt{L}(z) \lor \neg \mathtt{B}(z) \lor \neg \mathtt{S}(u) \lor \neg \mathtt{A}(u, z)) \land$$
$$\mathtt{L}(\mathtt{e}) \land \mathtt{B}(\mathtt{e})]$$

Recap

Goal:

- There is a student that is awake during all lectures.
- During all boring lectures no student keeps awake.

$\Rightarrow$ Then there are no boring lectures.

Steps:

- **Specify** the goal in predicate logic.
- **Negate** it.
- Transform the formula into **Prenex normal form**
- Use **Skolemization** to replace existential variables by function symbols
- Use **resolution** to obtain a contradiction.

Recap

Steps:

- **Specify** the goal in predicate logic.
- **Negate** it.
- Transform the formula into **Prenex normal form**
- Use **Skolemization**
- Use **resolution** to obtain a contradiction.

1	$\mathtt{S}(\mathtt{a})$	
2	$\neg\mathtt{L}(y) \vee \mathtt{A}(\mathtt{a}, y)$	
3	$\neg\mathtt{L}(z) \vee \neg\mathtt{B}(z) \vee \neg\mathtt{S}(u) \vee \neg\mathtt{A}(u, z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\mathtt{A}(\mathtt{a}, \mathtt{e})$	$(2, 4)$
7	$\neg\mathtt{B}(\mathtt{e}) \vee \neg\mathtt{S}(u) \vee \neg\mathtt{A}(u, \mathtt{e})$	$(3, 4)$
8	$\neg\mathtt{S}(u) \vee \neg\mathtt{A}(u, \mathtt{e})$	$(5, 7)$
9	$\neg\mathtt{A}(\mathtt{a}, \mathtt{e})$	$(1, 8)$
	$\perp$	$(6, 9)$

Recall: models in predicate logic

Idea:

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols
  - $[\mathtt{f}] : \mathcal{M}^n \to \mathcal{M}$ for every function symbol with arity n
  - $[\mathtt{P}] : \mathcal{M}^n \to \{\textit{false}, \textit{true}\}$ for every relation symbol with arity n

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols
  - $[\mathtt{f}] : \mathcal{M}^n \to \mathcal{M}$ for every function symbol with arity n
  - $[\mathtt{P}] : \mathcal{M}^n \to \{\textit{false}, \textit{true}\}$ for every relation symbol with arity n
- give a meaning to terms and formulas accordingly

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols
  - $[\mathtt{f}] : \mathcal{M}^n \to \mathcal{M}$ for every function symbol with arity n
  - $[\mathtt{P}] : \mathcal{M}^n \to \{\textit{false}, \textit{true}\}$ for every relation symbol with arity n
- give a meaning to terms and formulas accordingly
  - given: $\alpha : \mathcal{X} \to \mathcal{M}$
  - $[\![s]\!]_\alpha$ maps terms to $\mathcal{M}$
  - $[\![P]\!]_\alpha$ maps formulas to $\{\textit{false}, \textit{true}\}$

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols
  - $[\mathtt{f}] : \mathcal{M}^n \to \mathcal{M}$ for every function symbol with arity n
  - $[\mathtt{P}] : \mathcal{M}^n \to \{\mathit{false}, \mathit{true}\}$ for every relation symbol with arity n
- give a meaning to terms and formulas accordingly
  - given: $\alpha : \mathcal{X} \to \mathcal{M}$
  - $[\![s]\!]_\alpha$ maps terms to $\mathcal{M}$
  - $[\![P]\!]_\alpha$ maps formulas to $\{\mathit{false}, \mathit{true}\}$
- Let $\mathcal{M} \Vdash P$ if $[\![P]\!]_\alpha = \mathit{true}$ for any α

Recall: models in predicate logic

Idea:

- give a meaning to function symbols and relation symbols
  - $[\mathtt{f}] : \mathcal{M}^n \to \mathcal{M}$ for every function symbol with arity n
  - $[\mathtt{P}] : \mathcal{M}^n \to \{\textit{false}, \textit{true}\}$ for every relation symbol with arity n
- give a meaning to terms and formulas accordingly
  - given: $\alpha : \mathcal{X} \to \mathcal{M}$
  - $[\![s]\!]_\alpha$ maps terms to $\mathcal{M}$
  - $[\![P]\!]_\alpha$ maps formulas to $\{\textit{false}, \textit{true}\}$
- Let $\mathcal{M} \Vdash P$ if $[\![P]\!]_\alpha = \textit{true}$ for any α
- A formula P is **valid** if $\mathcal{M} \Vdash P$ for all models $\mathcal{M}$

Refutation-completeness

Theorem

(refutation-completeness of resolution)

A predicate in CNF is equivalent to *false* if and only if there is a sequence of resolution steps ending in the empty clause.

Refutation-completeness

Theorem

(refutation-completeness of resolution)

A predicate in CNF is equivalent to *false* if and only if there is a sequence of resolution steps ending in the empty clause.

That is:

there is a sequence of resolution steps ending in the empty clause

$$\Longleftrightarrow$$

there is no **model** $\mathcal{M}$ with $\mathcal{M} \Vdash P$

Proof outline

Let $\mathcal{X}$ be a set of clauses.

Proof outline

Let $\mathcal{X}$ be a set of clauses.

($\Leftarrow$) If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

Proof outline

Let $\mathcal{X}$ be a set of clauses.

($\Leftarrow$) If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

($\Rightarrow$) If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

- Suppose there is a model $\mathcal{M} := (M, [\![\cdot]\!]_\alpha)$ that makes all clauses in $\mathcal{X}$ true.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

- Suppose there is a model $\mathcal{M} := (M, [\![\cdot]\!]_\alpha)$ that makes all clauses in $\mathcal{X}$ true.
- Denote $\mathcal{M} \Vdash C$ if $[\![C]\!]_\alpha =$ *true* for all valuations α.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

- Suppose there is a model $\mathcal{M} := (M, [\![\cdot]\!]_\alpha)$ that makes all clauses in $\mathcal{X}$ true.
- Denote $\mathcal{M} \Vdash C$ if $[\![C]\!]_\alpha = \mathit{true}$ for all valuations α.
- If $\mathcal{M} \Vdash C$ then also $\mathcal{M} \Vdash C\sigma$ for all substitutions.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

- Suppose there is a model $\mathcal{M} := (M, [\![\cdot]\!]_\alpha)$ that makes all clauses in $\mathcal{X}$ true.
- Denote $\mathcal{M} \Vdash C$ if $[\![C]\!]_\alpha = \mathit{true}$ for all valuations α.
- If $\mathcal{M} \Vdash C$ then also $\mathcal{M} \Vdash C\sigma$ for all substitutions.
- Hence, if $\mathcal{M} \Vdash P \vee V$ and $\mathcal{M} \Vdash \neg Q \vee W$ and $P\tau = Q\tau$, then $\mathcal{M} \Vdash V\tau \vee W\tau$.

$\Longleftarrow$

Goal: If there is a sequence of resolution steps ending in the empty clause
then there is no model that makes all clauses in $\mathcal{X}$ true.

- Suppose there is a model $\mathcal{M} := (M, [\![\cdot]\!]_\alpha)$ that makes all clauses in $\mathcal{X}$ true.
- Denote $\mathcal{M} \Vdash C$ if $[\![C]\!]_\alpha = \mathit{true}$ for all valuations α.
- If $\mathcal{M} \Vdash C$ then also $\mathcal{M} \Vdash C\sigma$ for all substitutions.
- Hence, if $\mathcal{M} \Vdash P \vee V$ and $\mathcal{M} \Vdash \neg Q \vee W$ and $P\tau = Q\tau$, then $\mathcal{M} \Vdash V\tau \vee W\tau$.
- So truth is preserved under resolution.
  If we conclude $\perp$ (for which no model exists), then there was no model for the premises!

$$\Longrightarrow$$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

$\Longrightarrow$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

Proof: By constructing a model!

$$\Longrightarrow$$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

Proof: By constructing a model!

The basic idea: $\mathcal{M}$ is a set of *ground terms* (that is, terms without variables):

$\Longrightarrow$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

Proof: By constructing a model!

The basic idea: $\mathcal{M}$ is a set of *ground terms* (that is, terms without variables):

- infinitely many constants $\mathtt{c}_1, \mathtt{c}_2, \ldots$ are in $\mathcal{M}$;

$$\Longrightarrow$$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

Proof: By constructing a model!

The basic idea: $\mathcal{M}$ is a set of *ground terms* (that is, terms without variables):

- infinitely many constants $\mathtt{c}_1, \mathtt{c}_2, \ldots$ are in $\mathcal{M}$;
- if $\mathtt{f}$ is a function (not predicate!) of arity n, and $s_1, \ldots, s_n \in \mathcal{M}$, then $\mathtt{f}(s_1, \ldots, s_n) \in \mathcal{M}$.

$\Longrightarrow$

Goal: If there is no sequence of resolution steps ending in the empty clause
then there is a model that makes all clauses in $\mathcal{X}$ true.

Proof: By constructing a model!

The basic idea: $\mathcal{M}$ is a set of *ground terms* (that is, terms without variables):

- infinitely many constants $\mathtt{c}_1, \mathtt{c}_2, \ldots$ are in $\mathcal{M}$;
- if $\mathtt{f}$ is a function (not predicate!) of arity n, and $s_1, \ldots, s_n \in \mathcal{M}$, then $\mathtt{f}(s_1, \ldots, s_n) \in \mathcal{M}$.

We choose the value of all atomic formulas $\mathtt{P}(s_1, \ldots, s_n)$ step by step, to avoid a contradiction.

$$\Longrightarrow$$

Preparation:

$\Longrightarrow$

Preparation:

- Start with $\mathcal{M} := \emptyset$.

$\Longrightarrow$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$

$$\Longrightarrow$$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$
- Choose a **well-founded, total** ordering $\succ$ on ground terms.

$\Longrightarrow$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$
- Choose a **well-founded, total** ordering $\succ$ on ground terms. (For example: lexicographic comparison)

$\Longrightarrow$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$
- Choose a **well-founded, total** ordering $\succ$ on ground terms. (For example: lexicographic comparison)

Result: we obtain a total ordering on **ground clauses**.

$\Longrightarrow$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$
- Choose a **well-founded, total** ordering $\succ$ on ground terms. (For example: lexicographic comparison)

Result: we obtain a total ordering on **ground clauses.**

Idea of the proof: we go over all ground clauses $(\neg)A_1 \vee \cdots \vee (\neg)A_n$ in $\mathcal{N}$, in order of $\succ$

- If the clause is true in $\mathcal{M}$ as it is, do nothing.
- Otherwise, include the largest A_i into $\mathcal{M}$.

$$\Longrightarrow$$

Preparation:

- Start with $\mathcal{M} := \emptyset$.
- Let $\mathcal{N} := \{$all ground instances of clauses that can be obtained from $\mathcal{X}$ using resolution$\}$
- Choose a **well-founded, total** ordering $\succ$ on ground terms. (For example: lexicographic comparison)

Result: we obtain a total ordering on **ground clauses.**

Idea of the proof: we go over all ground clauses $(\neg)A_1 \vee \cdots \vee (\neg)A_n$ in $\mathcal{N}$, in order of $\succ$

- If the clause is true in $\mathcal{M}$ as it is, do nothing.
- Otherwise, include the largest A_i into $\mathcal{M}$.

The tricky part: the largest A_i does not occur negatively since $\mathcal{N}$ is closed under resolution!

Restricting resolution

A critical insight from the proof:

Restricting resolution

A critical insight from the proof:

- $Q = A \vee R_1 \vee \cdots \vee R_m$ with $A \succ_L R_i$ for all i
- $P = \neg A \vee L_1 \vee \cdots \vee L_n$ with $A \succ L_1 \vee \cdots \vee L_n$

Restricting resolution

A critical insight from the proof:

- $Q = A \lor R_1 \lor \cdots \lor R_m$ with $A \succ_L R_i$ for all i
- $P = \neg A \lor L_1 \lor \cdots \lor L_n$ with $A \succ L_1 \lor \cdots \lor L_n$

This allows us to be more restrictive in how we use resolution!

Restricting resolution

A critical insight from the proof:

- $Q = A \vee R_1 \vee \cdots \vee R_m$ with $A \succ_L R_i$ for all i
- $P = \neg A \vee L_1 \vee \cdots \vee L_n$ with $A \succ L_1 \vee \cdots \vee L_n$

This allows us to be more restrictive in how we use resolution!

Hope:

$$\frac{P \vee V \qquad \neg Q \vee W}{V\tau \vee W\tau}$$

τ an mgu such that $P\tau = Q\tau$
$P \succ L$ for all literals L in V
$Q \succ L$ for all literals L in W

Restricting resolution

A critical insight from the proof:

- $Q = A \vee R_1 \vee \cdots \vee R_m$ with $A \succ_L R_i$ for all i
- $P = \neg A \vee L_1 \vee \cdots \vee L_n$ with $A \succ L_1 \vee \cdots \vee L_n$

This allows us to be more restrictive in how we use resolution!

Hope:

$$\frac{P \vee V \qquad \neg Q \vee W}{V\tau \vee W\tau}$$

τ an mgu such that $P\tau = Q\tau$
$P \succ L$ for all literals L in V
$Q \succ L$ for all literals L in W

Problem: the proof assumed **ground** terms.

Restricting resolution

A critical insight from the proof:

- $Q = A \vee R_1 \vee \cdots \vee R_m$ with $A \succ_L R_i$ for all i
- $P = \neg A \vee L_1 \vee \cdots \vee L_n$ with $A \succ L_1 \vee \cdots \vee L_n$

This allows us to be more restrictive in how we use resolution!

Hope:

$$\frac{P \vee V \qquad \neg Q \vee W}{V\tau \vee W\tau}$$

τ an mgu such that $P\tau = Q\tau$
$P \succ L$ for all literals L in V
$Q \succ L$ for all literals L in W

Problem: the proof assumed **ground** terms.

Solution: we need $\succ$ such that:

If $P \succ Q$ then $P\sigma \succ Q\sigma$ for substitutions σ.

Restricting resolution

To be precise, let $\succ$ be an ordering on predicates such that:

Restricting resolution

To be precise, let $\succ$ be an ordering on predicates such that:

- $\succ$ is **stable**: if $P \succ Q$ then $P\sigma \succ Q\sigma$ for all σ;

Restricting resolution

To be precise, let $\succ$ be an ordering on predicates such that:

- $\succ$ is **stable**: if $P \succ Q$ then $P\sigma \succ Q\sigma$ for all σ;
- $\succ$ is **well-founded**: there is no infinite decreasing sequence $s_1 \succ s_2 \succ \ldots$;

Restricting resolution

To be precise, let $\succ$ be an ordering on predicates such that:

- $\succ$ is **stable**: if $P \succ Q$ then $P\sigma \succ Q\sigma$ for all σ;
- $\succ$ is **well-founded**: there is no infinite decreasing sequence $s_1 \succ s_2 \succ \ldots$;
- $\succ$ is **ground-total**: if P, Q are ground, then $P \succ Q$ or $Q \succ P$ or $P = Q$.

Restricting resolution

To be precise, let $\succ$ be an ordering on predicates such that:

- $\succ$ is **stable**: if $P \succ Q$ then $P\sigma \succ Q\sigma$ for all σ;
- $\succ$ is **well-founded**: there is no infinite decreasing sequence $s_1 \succ s_2 \succ \ldots$;
- $\succ$ is **ground-total**: if P, Q are ground, then $P \succ Q$ or $Q \succ P$ or $P = Q$.

An example of such an ordering is LPO.

Ordered resolution

Let **ordered resolution** be given by:

Ordered resolution

Let **ordered resolution** be given by:

$$\frac{P \vee V \qquad \neg Q \vee W}{V\tau \vee W\tau}$$

τ an mgu such that $P\tau = Q\tau$
for every atom L occurring in V: $L \not\succ P$
for every atom L occurring in W: $L \not\succ Q$

Ordered resolution

Let **ordered resolution** be given by:

$$\frac{P \vee V \qquad \neg Q \vee W}{V\tau \vee W\tau}$$

τ an mgu such that $P\tau = Q\tau$
for every atom L occurring in V: $L \not\succ P$
for every atom L occurring in W: $L \not\succ Q$

Theorem

(refutation-completeness of ordered resolution)

A predicate in CNF is equivalent to *false* if and only if there is a sequence of ordered resolution steps ending in the empty clause.

Ordered resolution: example

Let's try it out!

1 $\mathtt{S}(\mathtt{a})$

2 $\neg\mathtt{L}(x) \vee \mathtt{A}(\mathtt{a}, x)$

3 $\neg\mathtt{L}(y) \vee \neg\mathtt{B}(y) \vee \neg\mathtt{S}(z) \vee \neg\mathtt{A}(z, y)$

4 $\mathtt{L}(\mathtt{e})$

5 $\mathtt{B}(\mathtt{e})$

Ordered resolution: example

Let's try it out!

$$
\begin{array}{ll}
1 & \mathtt{S}(\mathtt{a}) \\
2 & \neg\mathtt{L}(x) \vee \mathtt{A}(\mathtt{a}, x) \\
3 & \neg\mathtt{L}(y) \vee \neg\mathtt{B}(y) \vee \neg\mathtt{S}(z) \vee \neg\mathtt{A}(z, y) \\
4 & \mathtt{L}(\mathtt{e}) \\
5 & \mathtt{B}(\mathtt{e})
\end{array}
$$

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg \mathtt{L}(x)$
3	$\neg \mathtt{A}(z, y) \vee \neg \mathtt{B}(y) \vee \neg \mathtt{L}(y) \vee \neg \mathtt{S}(z)$
4	$\mathtt{L}(\mathtt{e})$
5	$\mathtt{B}(\mathtt{e})$

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$	
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg \mathtt{L}(x)$	
3	$\neg \mathtt{A}(z, y) \vee \neg \mathtt{B}(y) \vee \neg \mathtt{L}(y) \vee \neg \mathtt{S}(z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\neg \mathtt{L}(y) \vee \neg \mathtt{B}(y) \vee \neg \mathtt{L}(y) \vee \neg \mathtt{S}(\mathtt{a})$	$(2, 3)$

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$	
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg \mathtt{L}(x)$	
3	$\neg \mathtt{A}(z, y) \vee \neg \mathtt{B}(y) \vee \neg \mathtt{L}(y) \vee \neg \mathtt{S}(z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\neg \mathtt{B}(y) \vee \neg \mathtt{L}(y) \vee \neg \mathtt{S}(\mathtt{a})$	$(2, 3)$

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$	
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg\mathtt{L}(x)$	
3	$\neg\mathtt{A}(z, y) \vee \neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(\mathtt{a})$	$(2, 3)$
7	$\neg\mathtt{L}(\mathtt{e}) \vee \neg\mathtt{S}(\mathtt{a})$	$(5, 6)$

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$	
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg\mathtt{L}(x)$	
3	$\neg\mathtt{A}(z, y) \vee \neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(\mathtt{a})$	$(2, 3)$
7	$\neg\mathtt{L}(\mathtt{e}) \vee \neg\mathtt{S}(\mathtt{a})$	$(5, 6)$
8	$\neg\mathtt{S}(\mathtt{a})$	$(4, 7)$

Ordered resolution: example

We arbitrarily set: $\mathtt{A} \rhd \mathtt{B} \rhd \mathtt{L} \rhd \mathtt{S}$.

1	$\mathtt{S}(\mathtt{a})$	
2	$\mathtt{A}(\mathtt{a}, x) \vee \neg\mathtt{L}(x)$	
3	$\neg\mathtt{A}(z, y) \vee \neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(z)$	
4	$\mathtt{L}(\mathtt{e})$	
5	$\mathtt{B}(\mathtt{e})$	
6	$\neg\mathtt{B}(y) \vee \neg\mathtt{L}(y) \vee \neg\mathtt{S}(\mathtt{a})$	$(2,3)$
7	$\neg\mathtt{L}(\mathtt{e}) \vee \neg\mathtt{S}(\mathtt{a})$	$(5,6)$
8	$\neg\mathtt{S}(\mathtt{a})$	$(4,7)$
9	$\perp$	$(1,8)$

Horn Clauses

Horn Clauses

A **Horn clause** is a clause with at most one positive literal.

Horn Clauses

A **Horn clause** is a clause with at most one positive literal.

Usually a Horn clause $C \vee \neg C_1 \vee \cdots \vee \neg C_n$ is written as

$$C \leftarrow C_1, \ldots, C_n$$

or as

$$C \;\; \texttt{:-} \;\; C_1, \ldots, C_n.$$

Horn Clauses

A **Horn clause** is a clause with at most one positive literal.

Usually a Horn clause $C \vee \neg C_1 \vee \cdots \vee \neg C_n$ is written as

$$C \leftarrow C_1, \ldots, C_n$$

or as

$$C \quad \texttt{:-} \quad C_1, \ldots, C_n.$$

The positive literal C is called the **head**.

Horn Clauses

A **Horn clause** is a clause with at most one positive literal.

Usually a Horn clause $C \vee \neg C_1 \vee \cdots \vee \neg C_n$ is written as

$$C \leftarrow C_1, \ldots, C_n$$

or as

$$C \quad \texttt{:-} \quad C_1, \ldots, C_n.$$

The positive literal C is called the **head**.

A clause without a head is called a **goal**.

Horn Clauses

A **Horn clause** is a clause with at most one positive literal.

Usually a Horn clause $C \vee \neg C_1 \vee \cdots \vee \neg C_n$ is written as

$$C \leftarrow C_1, \ldots, C_n$$

or as

$$C \quad \texttt{:-} \quad C_1, \ldots, C_n.$$

The positive literal C is called the **head**.

A clause without a head is called a **goal**.

A clause consisting only of a head is called a **fact**;
it is written as `C.` instead of `C :- .`

Resolution between Horn clauses

Resolution between Horn clauses

$$\frac{P \vee \neg V_1 \vee \cdots \vee \neg V_n \qquad Q \vee \neg P' \vee \neg W_1 \vee \ldots \neg \vee W_m}{(Q \vee \neg W_1 \vee \cdots \vee \neg W_m \vee \neg V_1 \vee \ldots \neg \vee V_n)\sigma} \quad \begin{array}{c}\sigma \text{ the mgu} \\ \text{of } P, P'\end{array}$$

Resolution between Horn clauses

$$\frac{P \vee \neg V_1 \vee \dots \vee \neg V_n \qquad Q \vee \neg P' \vee \neg W_1 \vee \dots \neg \vee W_m}{(Q \vee \neg W_1 \vee \dots \vee \neg W_m \vee \neg V_1 \vee \dots \neg \vee V_n)\sigma} \quad \sigma \text{ the mgu of } P, P'$$

A **Prolog program** is a set of Horn clauses in this notation.

Example

```
arrow(a,b).
arrow(a,c).
arrow(b,c).
arrow(c,d).
path(X,Y) :- arrow(X,Y).
path(X,Y) :- arrow(X,Z),path(Z,Y).
```

Example

```
arrow(a,b).
arrow(a,c).
arrow(b,c).
arrow(c,d).
path(X,Y) :- arrow(X,Y).
path(X,Y) :- arrow(X,Z),path(Z,Y).
```

The first four clauses define a directed graph on four nodes.

Example

```
arrow(a,b).
arrow(a,c).
arrow(b,c).
arrow(c,d).
path(X,Y) :- arrow(X,Y).
path(X,Y) :- arrow(X,Z),path(Z,Y).
```

The first four clauses define a directed graph on four nodes.

By the last two clauses the notion of a path in a graph is defined.

Example

```
arrow(a,b).
arrow(a,c).
arrow(b,c).
arrow(c,d).
path(X,Y) :- arrow(X,Y).
path(X,Y) :- arrow(X,Z),path(Z,Y).
```

The first four clauses define a directed graph on four nodes.

By the last two clauses the notion of a path in a graph is defined.

Question: does a path exist from `a` to `d`?

```
:- path(a,d)
```

Example

```
arrow(a,b).
arrow(a,c).
arrow(b,c).
arrow(c,d).
path(X,Y) :- arrow(X,Y).
path(X,Y) :- arrow(X,Z),path(Z,Y).
```

The first four clauses define a directed graph on four nodes.

By the last two clauses the notion of a path in a graph is defined.

Question: does a path exist from `a` to `d`?

```
:- path(a,d)
```

Answer: using resolution!

Resolution for Horn clauses

1	$\mathtt{arrow}(\mathtt{a}, \mathtt{b})$
2	$\mathtt{arrow}(\mathtt{a}, \mathtt{c})$
3	$\mathtt{arrow}(\mathtt{b}, \mathtt{c})$
4	$\mathtt{arrow}(\mathtt{c}, \mathtt{d})$
5	$\mathtt{path}(x, y) \vee \neg\mathtt{arrow}(x, y)$
6	$\mathtt{path}(x, y) \vee \neg\mathtt{arrow}(x, z) \vee \neg\mathtt{path}(z, y)$
7	$\neg\mathtt{path}(\mathtt{a}, \mathtt{d})$

Resolution for Horn clauses

$$
\begin{array}{ll}
1 & \texttt{arrow}(\texttt{a}, \texttt{b}) \\
2 & \texttt{arrow}(\texttt{a}, \texttt{c}) \\
3 & \texttt{arrow}(\texttt{b}, \texttt{c}) \\
4 & \texttt{arrow}(\texttt{c}, \texttt{d}) \\
5 & \texttt{path}(x, y) \vee \neg\texttt{arrow}(x, y) \\
6 & \texttt{path}(x, y) \vee \neg\texttt{arrow}(x, z) \vee \neg\texttt{path}(z, y) \\
7 & \neg\texttt{path}(\texttt{a}, \texttt{d}) \\
\hline
8 & \neg\texttt{arrow}(\texttt{a}, z) \vee \neg\texttt{path}(z, \texttt{d}) \quad (6, 7)
\end{array}
$$

Resolution for Horn clauses

1	$\texttt{arrow}(\texttt{a}, \texttt{b})$	
2	$\texttt{arrow}(\texttt{a}, \texttt{c})$	
3	$\texttt{arrow}(\texttt{b}, \texttt{c})$	
4	$\texttt{arrow}(\texttt{c}, \texttt{d})$	
5	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, y)$	
6	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, z) \vee \neg\texttt{path}(z, y)$	
7	$\neg\texttt{path}(\texttt{a}, \texttt{d})$	
8	$\neg\texttt{arrow}(\texttt{a}, z) \vee \neg\texttt{path}(z, \texttt{d})$	$(6, 7)$
9	$\neg\texttt{path}(\texttt{b}, \texttt{d})$	$(1, 8)$

Resolution for Horn clauses

1	$\mathtt{arrow}(\mathtt{a},\mathtt{b})$	
2	$\mathtt{arrow}(\mathtt{a},\mathtt{c})$	
3	$\mathtt{arrow}(\mathtt{b},\mathtt{c})$	
4	$\mathtt{arrow}(\mathtt{c},\mathtt{d})$	
5	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,y)$	
6	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,z) \vee \neg\mathtt{path}(z,y)$	
7	$\neg\mathtt{path}(\mathtt{a},\mathtt{d})$	
8	$\neg\mathtt{arrow}(\mathtt{a},z) \vee \neg\mathtt{path}(z,\mathtt{d})$	$(6,7)$
9	$\neg\mathtt{path}(\mathtt{b},\mathtt{d})$	$(1,8)$
10	$\neg\mathtt{arrow}(\mathtt{b},z) \vee \neg\mathtt{path}(z,\mathtt{d})$	$(6,9)$

Resolution for Horn clauses

1	$\texttt{arrow}(\texttt{a}, \texttt{b})$	
2	$\texttt{arrow}(\texttt{a}, \texttt{c})$	
3	$\texttt{arrow}(\texttt{b}, \texttt{c})$	
4	$\texttt{arrow}(\texttt{c}, \texttt{d})$	
5	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, y)$	
6	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, z) \vee \neg\texttt{path}(z, y)$	
7	$\neg\texttt{path}(\texttt{a}, \texttt{d})$	
8	$\neg\texttt{arrow}(\texttt{a}, z) \vee \neg\texttt{path}(z, \texttt{d})$	$(6, 7)$
9	$\neg\texttt{path}(\texttt{b}, \texttt{d})$	$(1, 8)$
10	$\neg\texttt{arrow}(\texttt{b}, z) \vee \neg\texttt{path}(z, \texttt{d})$	$(6, 9)$
11	$\neg\texttt{path}(\texttt{c}, \texttt{d})$	$(3, 10)$

Resolution for Horn clauses

1	$\texttt{arrow}(\texttt{a}, \texttt{b})$	
2	$\texttt{arrow}(\texttt{a}, \texttt{c})$	
3	$\texttt{arrow}(\texttt{b}, \texttt{c})$	
4	$\texttt{arrow}(\texttt{c}, \texttt{d})$	
5	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, y)$	
6	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, z) \vee \neg\texttt{path}(z, y)$	
7	$\neg\texttt{path}(\texttt{a}, \texttt{d})$	
8	$\neg\texttt{arrow}(\texttt{a}, z) \vee \neg\texttt{path}(z, \texttt{d})$	$(6, 7)$
9	$\neg\texttt{path}(\texttt{b}, \texttt{d})$	$(1, 8)$
10	$\neg\texttt{arrow}(\texttt{b}, z) \vee \neg\texttt{path}(z, \texttt{d})$	$(6, 9)$
11	$\neg\texttt{path}(\texttt{c}, \texttt{d})$	$(3, 10)$
12	$\neg\texttt{arrow}(\texttt{c}, \texttt{d})$	$(5, 11)$

Resolution for Horn clauses

1	$\mathtt{arrow}(\mathtt{a},\mathtt{b})$	
2	$\mathtt{arrow}(\mathtt{a},\mathtt{c})$	
3	$\mathtt{arrow}(\mathtt{b},\mathtt{c})$	
4	$\mathtt{arrow}(\mathtt{c},\mathtt{d})$	
5	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,y)$	
6	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,z) \vee \neg\mathtt{path}(z,y)$	
7	$\neg\mathtt{path}(\mathtt{a},\mathtt{d})$	
8	$\neg\mathtt{arrow}(\mathtt{a},z) \vee \neg\mathtt{path}(z,\mathtt{d})$	$(6,7)$
9	$\neg\mathtt{path}(\mathtt{b},\mathtt{d})$	$(1,8)$
10	$\neg\mathtt{arrow}(\mathtt{b},z) \vee \neg\mathtt{path}(z,\mathtt{d})$	$(6,9)$
11	$\neg\mathtt{path}(\mathtt{c},\mathtt{d})$	$(3,10)$
12	$\neg\mathtt{arrow}(\mathtt{c},\mathtt{d})$	$(5,11)$
13	$\perp$	$(4,12)$

Ordered resolution for Horn clauses

Choose: $\texttt{path} \rhd \texttt{arrow}$:

1	$\texttt{arrow}(\texttt{a}, \texttt{b})$
2	$\texttt{arrow}(\texttt{a}, \texttt{c})$
3	$\texttt{arrow}(\texttt{b}, \texttt{c})$
4	$\texttt{arrow}(\texttt{c}, \texttt{d})$
5	$\texttt{path}(x, y) \vee \neg\texttt{arrow}(x, y)$
6	$\texttt{path}(x, y) \vee \neg\texttt{path}(z, y) \vee \neg\texttt{arrow}(x, z)$
7	$\neg\texttt{path}(\texttt{a}, \texttt{d})$

Ordered resolution for Horn clauses

Choose: $\mathtt{path} \rhd \mathtt{arrow}$:

1	$\mathtt{arrow}(\mathtt{a}, \mathtt{b})$	
2	$\mathtt{arrow}(\mathtt{a}, \mathtt{c})$	
3	$\mathtt{arrow}(\mathtt{b}, \mathtt{c})$	
4	$\mathtt{arrow}(\mathtt{c}, \mathtt{d})$	
5	$\mathtt{path}(x, y) \vee \neg\mathtt{arrow}(x, y)$	
6	$\mathtt{path}(x, y) \vee \neg\mathtt{path}(z, y) \vee \neg\mathtt{arrow}(x, z)$	
7	$\neg\mathtt{path}(\mathtt{a}, \mathtt{d})$	
8	$\neg\mathtt{path}(z, \mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a}, z)$	$(6, 7)$

Ordered resolution for Horn clauses

Choose: $\mathtt{path} \rhd \mathtt{arrow}$:

1	$\mathtt{arrow}(\mathtt{a}, \mathtt{b})$
2	$\mathtt{arrow}(\mathtt{a}, \mathtt{c})$
3	$\mathtt{arrow}(\mathtt{b}, \mathtt{c})$
4	$\mathtt{arrow}(\mathtt{c}, \mathtt{d})$
5	$\mathtt{path}(x, y) \vee \neg\mathtt{arrow}(x, y)$
6	$\mathtt{path}(x, y) \vee \neg\mathtt{path}(z, y) \vee \neg\mathtt{arrow}(x, z)$
7	$\neg\mathtt{path}(\mathtt{a}, \mathtt{d})$
8	$\neg\mathtt{path}(z, \mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a}, z)$ $\quad (6, 7)$
9	$\neg\mathtt{arrow}(z, \mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a}, z)$ $\quad (5, 8)$

Ordered resolution for Horn clauses

Choose: $\mathtt{path} \rhd \mathtt{arrow}$:

1	$\mathtt{arrow}(\mathtt{a},\mathtt{b})$	
2	$\mathtt{arrow}(\mathtt{a},\mathtt{c})$	
3	$\mathtt{arrow}(\mathtt{b},\mathtt{c})$	
4	$\mathtt{arrow}(\mathtt{c},\mathtt{d})$	
5	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,y)$	
6	$\mathtt{path}(x,y) \vee \neg\mathtt{path}(z,y) \vee \neg\mathtt{arrow}(x,z)$	
7	$\neg\mathtt{path}(\mathtt{a},\mathtt{d})$	
8	$\neg\mathtt{path}(z,\mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a},z)$	$(6,7)$
9	$\neg\mathtt{arrow}(z,\mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a},z)$	$(5,8)$
10	$\neg\mathtt{arrow}(\mathtt{a},\mathtt{c})$	$(4,9)$

Ordered resolution for Horn clauses

Choose: $\mathtt{path} \rhd \mathtt{arrow}$:

1	$\mathtt{arrow}(\mathtt{a},\mathtt{b})$	
2	$\mathtt{arrow}(\mathtt{a},\mathtt{c})$	
3	$\mathtt{arrow}(\mathtt{b},\mathtt{c})$	
4	$\mathtt{arrow}(\mathtt{c},\mathtt{d})$	
5	$\mathtt{path}(x,y) \vee \neg\mathtt{arrow}(x,y)$	
6	$\mathtt{path}(x,y) \vee \neg\mathtt{path}(z,y) \vee \neg\mathtt{arrow}(x,z)$	
7	$\neg\mathtt{path}(\mathtt{a},\mathtt{d})$	
8	$\neg\mathtt{path}(z,\mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a},z)$	$(6,7)$
9	$\neg\mathtt{arrow}(z,\mathtt{d}) \vee \neg\mathtt{arrow}(\mathtt{a},z)$	$(5,8)$
10	$\neg\mathtt{arrow}(\mathtt{a},\mathtt{c})$	$(4,9)$
11	$\perp$	$(2,10)$

Prolog

This mechanism also applies for goals containing variables.

Prolog

This mechanism also applies for goals containing variables.

Refutation: `:- path(a,X)`
No refutation: `:- path(d,X)`

Prolog

This mechanism also applies for goals containing variables.

Refutation: `:- path(a,X)`
No refutation: `:- path(d,X)`

Prolog is a **declarative programming language**, it is a kind of **logic programming**.

Prolog

This mechanism also applies for goals containing variables.

Refutation: `:- path(a,X)`
No refutation: `:- path(d,X)`

Prolog is a **declarative programming language**, it is a kind of **logic programming**.

Optimisation: omit occur check (in unification)

```
p(X,X).
:- p(X,f(X)).
```

Prolog

This mechanism also applies for goals containing variables.

Refutation: `:- path(a,X)`
No refutation: `:- path(d,X)`

Prolog is a **declarative programming language**, it is a kind of **logic programming**.

Optimisation: omit occur check (in unification)

```
p(X,X).
:- p(X,f(X)).
```

$\Longrightarrow$ refutation-completeness is lost.

One critical feature still missing:

One critical feature still missing:

equality

One critical feature still missing:

equality

Usage: most applications of predicate logic!

One critical feature still missing:

equality

Usage: most applications of predicate logic!

- Reasoning over natural numbers:

$$\forall x[\forall y[\mathtt{suc}(x) = \mathtt{suc}(y) \to x = y]]$$

- Our students / lectures example:

$$\exists x[\exists y[x \neq y \wedge \mathtt{Favourite}(x) = \mathtt{AR} \wedge \mathtt{Favourite}(y) = \mathtt{AR}]]$$

Can we already do this?

Can we already do this?

$$\forall x[\mathtt{EQ}(x,x)]$$

$$\forall x\forall y[\mathtt{EQ}(x,y) \rightarrow \mathtt{EQ}(y,x)]$$

$$\forall x\forall y\forall z[\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(y,z) \rightarrow \mathtt{EQ}(x,z)]$$

Can we already do this?

$$\forall x[\mathtt{EQ}(x,x)]$$

$$\forall x \forall y[\mathtt{EQ}(x,y) \to \mathtt{EQ}(y,x)]$$

$$\forall x \forall y \forall z[\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(y,z) \to \mathtt{EQ}(x,z)]$$

$$\exists x[\exists y[\neg \mathtt{EQ}(x,y) \wedge \mathtt{EQ}(\mathtt{Favourite}(x), \mathtt{AR}) \wedge \mathtt{EQ}(\mathtt{Favourite}(y), \mathtt{AR})]]$$

Can we already do this?

$$\forall x[\mathtt{EQ}(x,x)]$$
$$\forall x \forall y[\mathtt{EQ}(x,y) \to \mathtt{EQ}(y,x)]$$
$$\forall x \forall y \forall z[\mathtt{EQ}(x,y) \land \mathtt{EQ}(y,z) \to \mathtt{EQ}(x,z)]$$

$$\exists x[\exists y[\neg \mathtt{EQ}(x,y) \land \mathtt{EQ}(\mathtt{Favourite}(x),\mathtt{AR}) \land \mathtt{EQ}(\mathtt{Favourite}(y),\mathtt{AR})]]$$

This is not sufficient!

Can we already do this?

$$\forall x[\mathtt{EQ}(x,x)]$$
$$\forall x\forall y[\mathtt{EQ}(x,y) \rightarrow \mathtt{EQ}(y,x)]$$
$$\forall x\forall y\forall z[\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(y,z) \rightarrow \mathtt{EQ}(x,z)]$$

$$\exists x[\exists y[\neg\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(\mathtt{Favourite}(x),\mathtt{AR}) \wedge \mathtt{EQ}(\mathtt{Favourite}(y),\mathtt{AR})]]$$

This is not sufficient!

$$\mathtt{P}(\mathtt{Alice}) \wedge \neg\mathtt{P}(\mathtt{Bob}) \wedge \mathtt{EQ}(\mathtt{Alice},\mathtt{Bob})$$

Can we already do this?

$$\forall x[\mathtt{EQ}(x,x)]$$
$$\forall x\forall y[\mathtt{EQ}(x,y) \rightarrow \mathtt{EQ}(y,x)]$$
$$\forall x\forall y\forall z[\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(y,z) \rightarrow \mathtt{EQ}(x,z)]$$

$$\exists x[\exists y[\neg\mathtt{EQ}(x,y)\wedge\mathtt{EQ}(\mathtt{Favourite}(x),\mathtt{AR})\wedge\mathtt{EQ}(\mathtt{Favourite}(y),\mathtt{AR})]]$$

This is not sufficient!

$$\mathtt{P}(\mathtt{Alice}) \wedge \neg\mathtt{P}(\mathtt{Bob}) \wedge \mathtt{EQ}(\mathtt{Alice},\mathtt{Bob})$$

And also:

$$\exists x[\mathtt{EQ}(x,\mathtt{Alice}) \wedge \neg\mathtt{EQ}(\mathtt{Favourite}(x),\mathtt{Favourite}(\mathtt{Alice}))]$$

$$\forall x[\mathtt{EQ}(x,x)]$$
$$\forall x \forall y[\mathtt{EQ}(x,y) \rightarrow \mathtt{EQ}(y,x)]$$
$$\forall x \forall y \forall z[\mathtt{EQ}(x,y) \land \mathtt{EQ}(y,z) \rightarrow \mathtt{EQ}(x,z)]$$

For all predicates $\mathtt{P}$ of arity n and all $i \in \{1,\ldots,n\}$:

$$\forall x_1 \ldots \forall x_n \forall y_i$$
$$[\mathtt{EQ}(x_i,y_i) \rightarrow (\mathtt{P}(x_1,\ldots,x_i,\ldots,x_n) \leftrightarrow \mathtt{P}(x_1,\ldots,y_i,\ldots,x_n))]$$

$$\forall x[\mathtt{EQ}(x,x)]$$
$$\forall x \forall y[\mathtt{EQ}(x,y) \to \mathtt{EQ}(y,x)]$$
$$\forall x \forall y \forall z[\mathtt{EQ}(x,y) \wedge \mathtt{EQ}(y,z) \to \mathtt{EQ}(x,z)]$$

For all predicates $\mathtt{P}$ of arity n and all $i \in \{1, \ldots, n\}$:

$$\forall x_1 \ldots \forall x_n \forall y_i$$
$$[\mathtt{EQ}(x_i, y_i) \to (\mathtt{P}(x_1, \ldots, x_i, \ldots, x_n) \leftrightarrow \mathtt{P}(x_1, \ldots, y_i, \ldots, x_n))]$$

For all functions $\mathtt{f}$ of arity n and all $i \in \{1, \ldots, n\}$:

$$\forall x_1 \ldots \forall x_n \forall y_i$$
$$[\mathtt{EQ}(x_i, y_i) \to \mathtt{EQ}(\mathtt{f}(x_1, \ldots, x_i, \ldots, x_n), \mathtt{f}(x_1, \ldots, y_i, \ldots, x_n)]$$

Supporting equality directly

Supporting equality directly

Recall: formula P in predicate logic is true in model $(M, [\![\cdot]\!]_\alpha)$ if $[\![P]\!]_\alpha =$ *true*.

Supporting equality directly

Recall: formula P in predicate logic is true in model $(M, [\![\cdot]\!]_\alpha)$ if $[\![P]\!]_\alpha =$ *true*.

To define truth in the extension, we simply add:

$$[\![s = t]\!]_\alpha = \textit{true} \text{ if and only if } [\![s]\!]_\alpha = [\![t]\!]_\alpha$$

Supporting equality directly

Recall: formula P in predicate logic is true in model $(M, [\![\cdot]\!]_\alpha)$ if $[\![P]\!]_\alpha =$ ***true***.

To define truth in the extension, we simply add:

$$[\![s = t]\!]_\alpha = \textit{true} \text{ if and only if } [\![s]\!]_\alpha = [\![t]\!]_\alpha$$

Transformation to Prenex normal form and Skolemization: unchanged!

Question: Do we need predicates other than $=$?

Question: Do we need predicates other than $=$?

Answer: No! Replace

$$\mathtt{P}(s_1, \ldots, s_n)$$

by

$$\mathtt{P}(s_1, \ldots, s_n) = \textit{true}$$

Question: Do we need predicates other than $=$?

Answer: No! Replace

$$\mathtt{P}(s_1, \ldots, s_n)$$

by

$$\mathtt{P}(s_1, \ldots, s_n) = \textit{true}$$

Hence we arrive at **equational logic**:

Formulas built from $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ *and atoms* $s = t$, *where* s *and* t *are terms (that may contain variables).*

Question: Do we need predicates other than $=$?

Answer: No! Replace

$$\mathtt{P}(s_1, \ldots, s_n)$$

by

$$\mathtt{P}(s_1, \ldots, s_n) = \textit{true}$$

Hence we arrive at **equational logic**:

Formulas built from $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ *and atoms* $s = t$*,*
where s *and* t *are terms (that may contain variables).*

Variables are implicitly universally quantified.

Question: Do we need predicates other than $=$?

Answer: No! Replace

$$\mathtt{P}(s_1, \ldots, s_n)$$

by

$$\mathtt{P}(s_1, \ldots, s_n) = \textit{true}$$

Hence we arrive at **equational logic**:

Formulas built from $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ and atoms $s = t$, where s and t are terms (that may contain variables).

Variables are implicitly universally quantified.

Functions (and constants) are implicitly existentially qualified.

Question: Do we need predicates other than $=$?

Answer: No! Replace

$$\mathtt{P}(s_1, \ldots, s_n)$$

by

$$\mathtt{P}(s_1, \ldots, s_n) = \textit{true}$$

Hence we arrive at **equational logic**:

Formulas built from $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$ *and atoms* $s = t$*,*
where s *and* t *are terms (that may contain variables).*

Variables are implicitly universally quantified.

Functions (and constants) are implicitly existentially qualified.

We have seen that predicate logic can be reduced to equational logic.

Term rewriting

Term rewriting

Rules can be seen as **oriented equations**.

Term rewriting

Rules can be seen as **oriented equations.**

The rules

$$\begin{array}{rcl} \mathtt{add}(\mathtt{0}, y) & \rightarrow & y \\ \mathtt{add}(\mathtt{s}(x), y) & \rightarrow & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

define the equations:

$$\begin{array}{lrcl} \forall y. & \mathtt{add}(\mathtt{0}, y) & = & y \\ \forall x \forall y. & \mathtt{add}(\mathtt{s}(x), y) & = & \mathtt{s}(\mathtt{add}(x, y)) \end{array}$$

Term rewriting

Theorem

Given equations $\mathcal{E} = \{\ell_1 = r_1, \ldots, \ldots, \ell_n = r_2\}$
And rules $\mathcal{R} = \{\ell_1 \to r_1, \ldots, \ell_n \to r_n\}$
For any model $(\mathcal{M}, [\![\cdot]\!]_\alpha)$ with $\mathcal{M} \Vdash \ell_i = r_i$ for all i:
If $s \to t$ then $[\![s]\!]_\alpha = [\![t]\!]_\alpha$ for all α.

Term rewriting

Theorem

Given equations $\mathcal{E} = \{\ell_1 = r_1, \ldots, \ldots, \ell_n = r_2\}$
And rules $\mathcal{R} = \{\ell_1 \to r_1, \ldots, \ell_n \to r_n\}$
For any model $(\mathcal{M}, [\![\cdot]\!]_\alpha)$ with $\mathcal{M} \Vdash \ell_i = r_i$ for all i:
If $s \to t$ then $[\![s]\!]_\alpha = [\![t]\!]_\alpha$ for all α.

Proof: By induction on the size of s.

Term rewriting

Theorem

Given equations $\mathcal{E} = \{\ell_1 = r_1, \ldots, \ldots, \ell_n = r_2\}$
And rules $\mathcal{R} = \{\ell_1 \to r_1, \ldots, \ell_n \to r_n\}$
For any model $(\mathcal{M}, [\![\cdot]\!]_\alpha)$ with $\mathcal{M} \Vdash \ell_i = r_i$ for all i:
If $s \to t$ then $[\![s]\!]_\alpha = [\![t]\!]_\alpha$ for all α.

Proof: By induction on the size of s.

Corollary: if $s \leftrightarrow_{\mathcal{R}} t$, then $s = t$ follows from equations in $\mathcal{E}$.

Question:

Given equations $\mathcal{E} = \{s_1 = t_1, \ldots, s_n = t_n\}$*,*
And given another equation $u = v$*.*
Does $u = v$ *follow from* $\mathcal{E}$*?*

Question:

Given equations $\mathcal{E} = \{s_1 = t_1, \ldots, s_n = t_n\}$*,*
And given another equation $u = v$*.*
Does $u = v$ *follow from* $\mathcal{E}$*?*

(For example: if $\mathcal{E}$ contains the axioms for addition, does it follow that always $\texttt{add}(x, y) = \texttt{add}(y, x)$?)

Question:

Given equations $\mathcal{E} = \{s_1 = t_1, \ldots, s_n = t_n\}$*,*
And given another equation $u = v$*.*
Does $u = v$ *follow from* $\mathcal{E}$*?*

(For example: if $\mathcal{E}$ contains the axioms for addition, does it follow that always $\mathtt{add}(x, y) = \mathtt{add}(y, x)$?)

Method:

Completion

Question:

Given equations $\mathcal{E} = \{s_1 = t_1, \ldots, s_n = t_n\}$,
And given another equation $u = v$.
Does $u = v$ *follow from* $\mathcal{E}$*?*

(For example: if $\mathcal{E}$ contains the axioms for addition, does it follow that always $\mathtt{add}(x, y) = \mathtt{add}(y, x)$?)

Method:

- Use completion to find a **terminating**, **confluent** TRS $\mathcal{R}$ such that $\leftrightarrow_{\mathcal{R}}$ is exactly $=_{\mathcal{E}}$.
- Then see if $u \downarrow_{\mathcal{R}}$ and $v \downarrow_{\mathcal{R}}$ are the same!

The general case

We have;

If $s_1 = t_1$ *and . . . and* $s_n = t_n$
then also $u = v$.

The general case

We have;

If $s_1 = t_1$ *and ... and* $s_n = t_n$
then also $u = v$.

General case:

If φ_1 *and ... and* φ_n
then also $u = v$.

where all φ_i are **clauses** $L_1 \vee \cdots \vee L_k$ with each L_i either an equality $s = t$ or an inequality $s \neq t$.

The general case

We have;

If $s_1 = t_1$ *and* ... *and* $s_n = t_n$
then also $u = v$.

General case:

If φ_1 *and* ... *and* φ_n
then also $u = v$.

where all φ_i are **clauses** $L_1 \vee \cdots \vee L_k$ with each L_i either an equality $s = t$ or an inequality $s \neq t$.

Put differently: we should be able to prove unsatisfiablity of:

$$\varphi_1 \wedge \cdots \wedge \varphi_n \wedge \psi$$

where ψ and all φ_i are clauses.

Resolution?

To use resolution we would have to include clauses like:

$$x = x$$

$$x \neq y \lor y = x$$

$$x \neq y \lor y \neq z \lor x = z$$

Resolution?

To use resolution we would have to include clauses like:

$$x = x$$

$$x \neq y \vee y = x$$

$$x \neq y \vee y \neq z \vee x = z$$

and

$$x_i \neq y_i \vee \mathtt{f}(x_1, \ldots, x_i, \ldots, x_n) = \mathtt{f}(x_1, \ldots, y_i, \ldots, x_n)$$

for all $\mathtt{f}$ and argument positions i.

Resolution?

To use resolution we would have to include clauses like:

$$x = x$$

$$x \neq y \vee y = x$$

$$x \neq y \vee y \neq z \vee x = z$$

and

$$x_i \neq y_i \vee \mathtt{f}(x_1, \ldots, x_i, \ldots, x_n) = \mathtt{f}(x_1, \ldots, y_i, \ldots, x_n)$$

for all $\mathtt{f}$ and argument positions i.

Better: dedicated methods!

Goal: a form of resolution for equational logic!

Goal: a form of resolution for equational logic!

What should hold? Let us consider ground equations:

Goal: a form of resolution for equational logic!

What should hold? Let us consider ground equations:

$$\frac{s = t \vee V \qquad s \neq t \vee W}{V \vee W}$$

Goal: a form of resolution for equational logic!

What should hold? Let us consider ground equations:

$$\frac{s = t \lor V \qquad s \neq t \lor W}{V \lor W}$$

But we'll also need:

$$\frac{s = t \lor V \qquad \mathtt{f}(\dots, s, \dots) \neq \mathtt{f}(\dots, t, \dots) \lor W}{V \lor W}$$

Goal: a form of resolution for equational logic!

What should hold? Let us consider ground equations:

$$\frac{s = t \vee V \qquad s \neq t \vee W}{V \vee W}$$

But we'll also need:

$$\frac{s = t \vee V \qquad \mathtt{f}(\ldots, s, \ldots) \neq \mathtt{f}(\ldots, t, \ldots) \vee W}{V \vee W}$$

We can combine this through a context.

$$\frac{s = t \vee V \qquad C[s] \neq C[t] \vee W}{V \vee W}$$

Reflexivity and symmetry

Reflexivity and symmetry

Reflexivity:

$$\frac{s \neq s \vee V}{V}$$

Reflexivity and symmetry

Reflexivity:

$$\frac{s \neq s \vee V}{V}$$

Symmetry:

$$\frac{s = t \vee V}{t = s \vee V} \qquad \frac{s \neq t \vee V}{t \neq s \vee V}$$

Reflexivity and symmetry

Reflexivity:

$$\frac{s \neq s \vee V}{V}$$

Symmetry:

$$\frac{s = t \vee V}{t = s \vee V} \qquad \frac{s \neq t \vee V}{t \neq s \vee V}$$

Better: we just agree that we can freely swap order when applying rules.

Transitivity

We should be able to derive:

Transitivity

We should be able to derive:

$$\frac{s = t \vee U \qquad t = q \vee V \qquad s \neq q \vee W}{U \vee V \vee W}$$

Transitivity

We should be able to derive:

$$\frac{s = t \vee U \qquad t = q \vee V \qquad s \neq q \vee W}{U \vee V \vee W}$$

And also:

$$\frac{s = t \vee V_1 \qquad \mathtt{f}(t) = \mathtt{f}(q) \vee V_2 \qquad \mathtt{f}(q) = \mathtt{f}(u) \vee V_3 \qquad \mathtt{g}(\mathtt{f}(s), x) \neq \mathtt{g}(\mathtt{f}(u), x) \vee W}{V_1 \vee V_2 \vee V_3 \vee W}$$

Transitivity

We should be able to derive:

$$\frac{s = t \lor U \qquad t = q \lor V \qquad s \neq q \lor W}{U \lor V \lor W}$$

And also:

$$\frac{s = t \lor V_1 \qquad \mathtt{f}(t) = \mathtt{f}(q) \lor V_2 \qquad \mathtt{f}(q) = \mathtt{f}(u) \lor V_3 \qquad \mathtt{g}(\mathtt{f}(s), x) \neq \mathtt{g}(\mathtt{f}(u), x) \lor W}{V_1 \lor V_2 \lor V_3 \lor W}$$

And also:

$$\frac{s = t \lor U \qquad u = v \lor V \qquad \mathtt{f}(s, u) \neq \mathtt{f}(t, v) \lor W}{U \lor V \lor W}$$

Transitivity

We should be able to derive:

$$\frac{s = t \vee U \qquad t = q \vee V \qquad s \neq q \vee W}{U \vee V \vee W}$$

And also:

$$\frac{s = t \vee V_1 \qquad \mathtt{f}(t) = \mathtt{f}(q) \vee V_2 \qquad \mathtt{f}(q) = \mathtt{f}(u) \vee V_3 \qquad \mathtt{g}(\mathtt{f}(s), x) \neq \mathtt{g}(\mathtt{f}(u), x) \vee W}{V_1 \vee V_2 \vee V_3 \vee W}$$

And also:

$$\frac{s = t \vee U \qquad u = v \vee V \qquad \mathtt{f}(s, u) \neq \mathtt{f}(t, v) \vee W}{U \vee V \vee W}$$

Idea:

$$\frac{s = t \vee V \qquad C[s] = q \vee W}{C[t] = q \vee V \vee W} \qquad \frac{s = t \vee V \qquad C[s] \neq q \vee W}{C[t] \neq q \vee V \vee W}$$

Observation: we don't need resolution anymore!

Observation: we don't need resolution anymore!

$$\frac{s = t \vee V \qquad C[s] \neq C[t] \vee W}{V \vee W}$$

Observation: we don't need resolution anymore!

$$\frac{s = t \lor V \qquad C[s] \neq C[t] \lor W}{V \lor W}$$

This can be derived in two steps by:

$$\frac{s = t \lor V \qquad C[s] \neq q \lor W}{C[t] \neq q \lor V \lor W} \qquad\qquad \frac{q \neq q \lor U}{U}$$

For $q = C[t]$ and $U = V \lor W$.

Equality factoring

It turns out we need one more rule:

Equality factoring

It turns out we need one more rule:

$$\frac{s = t \lor s = u \lor V}{s = t \lor t \neq u \lor V}$$

Overview (for **ground** equations)

Equality resolution:

$$\frac{s \neq s \vee V}{V}$$

Positive superposition

$$\frac{s = t \vee V \qquad C[s] = q \vee W}{C[t] = q \vee V \vee W}$$

Negative superposition

$$\frac{s = t \vee V \qquad C[s] \neq q \vee W}{C[t] \neq q \vee V \vee W}$$

Equality factoring

$$\frac{s = t \vee s = u \vee V}{s = t \vee t \neq u \vee V}$$

Superposition calculus for **general** equations

Superposition calculus for **general** equations

Equality resolution:

$$\frac{s \neq t \vee V}{V\sigma}\ \sigma = mgu(s,t)$$

Superposition calculus for **general** equations

Equality resolution:

$$\frac{s \neq t \vee V}{V\sigma}\ \sigma = mgu(s,t)$$

Positive superposition

$$\frac{s = t \vee V \qquad C[u] = q \vee W}{(C[t] = q \vee V \vee W)\sigma}\quad \begin{array}{l}\sigma = mgu(s,u) \text{ and} \\ u \text{ is not a variable}\end{array}$$

Superposition calculus for **general** equations

Equality resolution:

$$\frac{s \neq t \vee V}{V\sigma}\ \sigma = mgu(s,t)$$

Positive superposition

$$\frac{s = t \vee V \qquad C[u] = q \vee W}{(C[t] = q \vee V \vee W)\sigma}\quad \begin{array}{l}\sigma = mgu(s,u) \text{ and} \\ u \text{ is not a variable}\end{array}$$

Negative superposition

$$\frac{s = t \vee V \qquad C[u] \neq q \vee W}{(C[t] \neq q \vee V \vee W)\sigma}\quad \begin{array}{l}\sigma = mgu(s,u) \text{ and} \\ u \text{ is not a variable}\end{array}$$

Superposition calculus for **general** equations

Equality resolution:

$$\frac{s \neq t \vee V}{V\sigma} \quad \sigma = mgu(s,t)$$

Positive superposition

$$\frac{s = t \vee V \qquad C[u] = q \vee W}{(C[t] = q \vee V \vee W)\sigma} \quad \begin{array}{l}\sigma = mgu(s,u) \text{ and} \\ u \text{ is not a variable}\end{array}$$

Negative superposition

$$\frac{s = t \vee V \qquad C[u] \neq q \vee W}{(C[t] \neq q \vee V \vee W)\sigma} \quad \begin{array}{l}\sigma = mgu(s,u) \text{ and} \\ u \text{ is not a variable}\end{array}$$

Equality factoring

$$\frac{s = t \vee q = u \vee V}{(s = t \vee t \neq u \vee V)\sigma} \quad \sigma = mgu(s,q)$$

Example:

We want to prove:

Example:

We want to prove:

- If $\forall x, y[\texttt{leq}(x, y) \vee \texttt{leq}(y, x)]$
- and $\forall x, y[\texttt{leq}(x, y) \rightarrow \texttt{max}(x, y) = y]$
- and $\forall x, y[\texttt{leq}(y, x) \rightarrow \texttt{max}(x, y) = x]$
- and $\forall x, y, z[\texttt{max}(\texttt{max}(x, y), z) = \texttt{max}(x, \texttt{max}(y, z))]$
- then $\forall x, y, z[\texttt{leq}(x, y) \wedge \texttt{leq}(y, z) \rightarrow \texttt{leq}(x, z)]$

Example:

We want to prove:

- If $\forall x, y[\mathtt{leq}(x, y) \vee \mathtt{leq}(y, x)]$
- and $\forall x, y[\mathtt{leq}(x, y) \to \mathtt{max}(x, y) = y]$
- and $\forall x, y[\mathtt{leq}(y, x) \to \mathtt{max}(x, y) = x]$
- and $\forall x, y, z[\mathtt{max}(\mathtt{max}(x, y), z) = \mathtt{max}(x, \mathtt{max}(y, z))]$
- then $\forall x, y, z[\mathtt{leq}(x, y) \wedge \mathtt{leq}(y, z) \to \mathtt{leq}(x, z)]$

To do this, we will try to refute the **negation** of the above implication.

Example:

We want to prove:

- If $\forall x, y[\texttt{leq}(x, y) \vee \texttt{leq}(y, x)]$
- and $\forall x, y[\texttt{leq}(x, y) \rightarrow \texttt{max}(x, y) = y]$
- and $\forall x, y[\texttt{leq}(y, x) \rightarrow \texttt{max}(x, y) = x]$
- and $\forall x, y, z[\texttt{max}(\texttt{max}(x, y), z) = \texttt{max}(x, \texttt{max}(y, z))]$
- then $\forall x, y, z[\texttt{leq}(x, y) \wedge \texttt{leq}(y, z) \rightarrow \texttt{leq}(x, z)]$

To do this, we will try to refute the **negation** of the above implication.

We also replace predicate symbols by function symbols, and translate implications $a \rightarrow b$ by $\neg a \vee b$.

Example:

We want to prove:

$$
\begin{array}{lr}
\forall x, y[\mathtt{leq}(x, y) = \mathtt{true} \lor \mathtt{leq}(y, x) = \mathtt{true}] & \land \\
\forall x, y[\mathtt{leq}(x, y) \neq \mathtt{true} \lor \mathtt{max}(x, y) = y] & \land \\
\forall x, y[\mathtt{leq}(x, y) \neq \mathtt{true} \lor \mathtt{max}(x, y) = x] & \land \\
\forall x, y, z[\mathtt{max}(\mathtt{max}(x, y), z) = \mathtt{max}(x, \mathtt{max}(y, z))] & \land \\
\exists x, y, z[\mathtt{leq}(x, y) = \mathtt{true} \land \mathtt{leq}(y, z) = \mathtt{true} \land \mathtt{leq}(y, z) \neq \mathtt{true}] &
\end{array}
$$

To do this, we will try to refute the **negation** of the above implication.

We also replace predicate symbols by function symbols, and translate implications $a \to b$ by $\neg a \lor b$.

Example:

We want to prove:

$$
\begin{array}{lr}
\forall x, y[\mathtt{leq}(x, y) = \mathtt{true} \vee \mathtt{leq}(y, x) = \mathtt{true}] & \wedge \\
\forall x, y[\mathtt{leq}(x, y) \neq \mathtt{true} \vee \mathtt{max}(x, y) = y] & \wedge \\
\forall x, y[\mathtt{leq}(x, y) \neq \mathtt{true} \vee \mathtt{max}(x, y) = x] & \wedge \\
\forall x, y, z[\mathtt{max}(\mathtt{max}(x, y), z) = \mathtt{max}(x, \mathtt{max}(y, z))] & \wedge \\
\exists x, y, z[\mathtt{leq}(x, y) = \mathtt{true} \wedge \mathtt{leq}(y, z) = \mathtt{true} \wedge \mathtt{leq}(y, z) \neq \mathtt{true}] &
\end{array}
$$

To do this, we will try to refute the **negation** of the above implication.

We also replace predicate symbols by function symbols, and translate implications $a \rightarrow b$ by $\neg a \vee b$.

After Skolemization, we obtain a CNF which we can do superposition on!

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
8	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	$(5, 2, \text{neg.sup})$

1	$\texttt{leq}(x,y) = \texttt{true} \vee \texttt{leq}(y,x) = \texttt{true}$	
2	$\texttt{leq}(x,y) \neq \texttt{true} \vee \texttt{max}(x,y) = y$	
3	$\texttt{leq}(y,x) \neq \texttt{true} \vee \texttt{max}(x,y) = x$	
4	$\texttt{max}(\texttt{max}(x,y),z) = \texttt{max}(x,\texttt{max}(y,z))$	
5	$\texttt{leq}(\texttt{a},\texttt{b}) = \texttt{true}$	
6	$\texttt{leq}(\texttt{b},\texttt{c}) = \texttt{true}$	
7	$\texttt{leq}(\texttt{a},\texttt{c}) \neq \texttt{true}$	
8	$\texttt{true} \neq \texttt{true} \vee \texttt{max}(\texttt{a},\texttt{b}) = \texttt{b}$	$(5, 2,$ neg.sup$)$
9	$\texttt{max}(\texttt{a},\texttt{b}) = \texttt{b}$	$(9,$ eq.res$)$

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
10	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(6, 2, neg.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \lor \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \lor \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \lor \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	$(9, \text{eq.res})$
10	$\mathtt{true} \neq \mathtt{true} \lor \mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	$(6, 2, \text{neg.sup})$
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	$(10, \text{eq.res})$

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)

1	$\mathtt{leq}(x,y) = \mathtt{true} \lor \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \lor \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \lor \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \lor \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
16	$\mathtt{max}(\mathtt{b},z) = \mathtt{max}(\mathtt{a},\mathtt{max}(\mathtt{b},z))$	(9, 4, pos.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
16	$\mathtt{max}(\mathtt{b},z) = \mathtt{max}(\mathtt{a},\mathtt{max}(\mathtt{b},z))$	(9, 4, pos.sup)
17	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{max}(\mathtt{a},\mathtt{c})$	(11, 16, pos.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	$(9, \text{eq.res})$
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	$(10, \text{eq.res})$
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	$(1, 3, \text{neg.sup, eq.res})$
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	$(13, 7, \text{neg.sup, eq.res})$
16	$\mathtt{max}(\mathtt{b},z) = \mathtt{max}(\mathtt{a},\mathtt{max}(\mathtt{b},z))$	$(9, 4, \text{pos.sup})$
17	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{max}(\mathtt{a},\mathtt{c})$	$(11, 16, \text{pos.sup})$
18	$\mathtt{c} = \mathtt{max}(\mathtt{a},\mathtt{c})$	$(11, 17, \text{pos.sup})$

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
16	$\mathtt{max}(\mathtt{b},z) = \mathtt{max}(\mathtt{a},\mathtt{max}(\mathtt{b},z))$	(9, 4, pos.sup)
17	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{max}(\mathtt{a},\mathtt{c})$	(11, 16, pos.sup)
18	$\mathtt{c} = \mathtt{max}(\mathtt{a},\mathtt{c})$	(11, 17, pos.sup)
19	$\mathtt{c} = \mathtt{a}$	(15, 18, pos.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \lor \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \lor \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \lor \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \lor \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
19	$\mathtt{c} = \mathtt{a}$	(15, 18, pos.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
19	$\mathtt{c} = \mathtt{a}$	(15, 18, pos.sup)
20	$\mathtt{leq}(\mathtt{a},\mathtt{a}) \neq \mathtt{true}$	(19, 7, neg.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
19	$\mathtt{c} = \mathtt{a}$	(15, 18, pos.sup)
20	$\mathtt{leq}(\mathtt{a},\mathtt{a}) \neq \mathtt{true}$	(19, 7, neg.sup)
21	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{leq}(\mathtt{a},\mathtt{a}) = \mathtt{true}$	(1, 20, neg.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	(9, eq.res)
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	(10, eq.res)
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	(1, 3, neg.sup, eq.res)
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	(13, 7, neg.sup, eq.res)
19	$\mathtt{c} = \mathtt{a}$	(15, 18, pos.sup)
20	$\mathtt{leq}(\mathtt{a},\mathtt{a}) \neq \mathtt{true}$	(19, 7, neg.sup)
21	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{leq}(\mathtt{a},\mathtt{a}) = \mathtt{true}$	(1, 20, neg.sup)
22	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{true} \neq \mathtt{true}$	(20, 21, neg.sup)

1	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{leq}(y,x) = \mathtt{true}$	
2	$\mathtt{leq}(x,y) \neq \mathtt{true} \vee \mathtt{max}(x,y) = y$	
3	$\mathtt{leq}(y,x) \neq \mathtt{true} \vee \mathtt{max}(x,y) = x$	
4	$\mathtt{max}(\mathtt{max}(x,y),z) = \mathtt{max}(x,\mathtt{max}(y,z))$	
5	$\mathtt{leq}(\mathtt{a},\mathtt{b}) = \mathtt{true}$	
6	$\mathtt{leq}(\mathtt{b},\mathtt{c}) = \mathtt{true}$	
7	$\mathtt{leq}(\mathtt{a},\mathtt{c}) \neq \mathtt{true}$	
9	$\mathtt{max}(\mathtt{a},\mathtt{b}) = \mathtt{b}$	$(9, \text{eq.res})$
11	$\mathtt{max}(\mathtt{b},\mathtt{c}) = \mathtt{c}$	$(10, \text{eq.res})$
13	$\mathtt{leq}(x,y) = \mathtt{true} \vee \mathtt{max}(x,y) = x$	$(1, 3, \text{neg.sup, eq.res})$
15	$\mathtt{max}(\mathtt{a},\mathtt{c}) = \mathtt{a}$	$(13, 7, \text{neg.sup, eq.res})$
19	$\mathtt{c} = \mathtt{a}$	$(15, 18, \text{pos.sup})$
20	$\mathtt{leq}(\mathtt{a},\mathtt{a}) \neq \mathtt{true}$	$(19, 7, \text{neg.sup})$
21	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{leq}(\mathtt{a},\mathtt{a}) = \mathtt{true}$	$(1, 20, \text{neg.sup})$
22	$\mathtt{true} \neq \mathtt{true} \vee \mathtt{true} \neq \mathtt{true}$	$(20, 21, \text{neg.sup})$
23	$\perp$	$(22, \text{eq.res})$

Combining steps

Optimisation: let's keep resolution. :)

Combining steps

Optimisation: let's keep resolution. :)

$$\frac{s = t \vee V \qquad C[u] \neq q \vee W}{(V \vee W)\sigma}$$

$\sigma = mgu(s, u)$ and u is not a variable and $C[t]\sigma = q\sigma$

Combining steps

Optimisation: let's keep resolution. :)

$$\frac{s = t \vee V \qquad C[u] \neq q \vee W}{(V \vee W)\sigma}$$

$\sigma = mgu(s, u)$ and u is not a variable and $C[t]\sigma = q\sigma$

However: superposition is not typically done by hand!

Result

Theorem

refutation-completeness of superposition

A CNF of equational clauses is equivalent to *false* if and only if there is a sequence of superposition steps ending in the empty clause.

Result

Theorem

refutation-completeness of superposition

A CNF of equational clauses is equivalent to *false* if and only if there is a sequence of superposition steps ending in the empty clause.

Proof idea: same as for resolution!

Result

Theorem

refutation-completeness of superposition

A CNF of equational clauses is equivalent to *false* if and only if there is a sequence of superposition steps ending in the empty clause.

Proof idea: same as for resolution!

- Assume $\perp$ cannot be derived using superposition.

Result

Theorem

refutation-completeness of superposition

A CNF of equational clauses is equivalent to *false* if and only if there is a sequence of superposition steps ending in the empty clause.

Proof idea: same as for resolution!

- Assume $\perp$ cannot be derived using superposition.
- Use a ground total, well-founded ordering $\succ$ on ground terms.

Result

Theorem

refutation-completeness of superposition

A CNF of equational clauses is equivalent to *false* if and only if there is a sequence of superposition steps ending in the empty clause.

Proof idea: same as for resolution!

- Assume $\perp$ cannot be derived using superposition.
- Use a ground total, well-founded ordering $\succ$ on ground terms.
- Step-by-step (by induction on $\succ$), build a complete TRS.

Ordered superposition

Ordered superposition

- when using $s = t \vee V$ or $s \neq t \vee V$ in any of the rules except equality resolution, $t \not\succeq s$;
- when we consider a clause $s = t \vee V$ in one of the superposition rules, there is no literal L in V with $L \succ_L s = t$;
- when we consider a clause $s \neq t \vee V$ in one of the superposition rules, there is no literal L in V with $L \succeq_L s \neq t$;
- when using $s = t \vee V$ and $C[u] = q \vee W$ in positive superposition to conclude $(C[t] = q \vee V \vee W)\sigma$, not $s = t \succeq_L C[u] = q$;
- when using $s = t \vee V$ and $C[u] = q \vee W$ in negative superposition to conclude $(C[t] = q \vee V \vee W)\sigma$, not $s = t \succeq_L C[u] = q$.

Functional program analysis

```
let rec append a b =
    match a with
        | [] -> b
        | h ::  t -> h ::  append t b;;
```

Functional program analysis

```
let rec append a b =
    match a with
        | [] -> b
        | h ::  t -> h ::  append t b;;
```

corresponds exactly to the TRS rules:

$$\begin{aligned} \mathtt{append}(\mathtt{nil}, b) &\rightarrow b \\ \mathtt{append}(\mathtt{cons}(h, t)) &\rightarrow \mathtt{cons}(h, \mathtt{append}(t, b)) \end{aligned}$$

Struggles of functional program analysis

Additional features:

Struggles of functional program analysis

Additional features:

- integers, floating point numbers, etc.

Struggles of functional program analysis

Additional features:

- integers, floating point numbers, etc.
- higher-order functions (i.e., where a function can be passed as an argument)

Struggles of functional program analysis

Additional features:

- integers, floating point numbers, etc.
- higher-order functions (i.e., where a function can be passed as an argument)
- side effects such as storing values in a global variable

Struggles of functional program analysis

Additional features:

- integers, floating point numbers, etc.
- higher-order functions (i.e., where a function can be passed as an argument)
- side effects such as storing values in a global variable
- …

Analysing functional programs anyway:

Solutions:

Analysing functional programs anyway:

Solutions:

- **translate** functional programs to TRSs, while retaining some desirable properties

Analysing functional programs anyway:

Solutions:

- **translate** functional programs to TRSs, while retaining some desirable properties

```
let rec length l =
    match l with
        | [] -> 0
        | _ ::  t -> 1 + length t;;
```

$$
\begin{array}{rcl}
\mathtt{length}(\mathtt{nil}) & \to & \mathtt{0} \\
\mathtt{length}(\mathtt{cons}(x,t)) & \to & \mathtt{s}(\mathtt{length}(t))
\end{array}
$$

Analysing functional programs anyway:

Solutions:

- **translate** functional programs to TRSs, while retaining some desirable properties

```
let rec length l =
    match l with
        | [] -> 0
        | _ ::  t -> 1 + length t;;
```

$$\begin{array}{rcl} \mathtt{length}(\mathtt{nil}) & \to & \mathtt{0} \\ \mathtt{length}(\mathtt{cons}(x,t)) & \to & \mathtt{s}(\mathtt{length}(t)) \end{array}$$

- **transpose** methods from term rewriting to functional programs

Analysing functional programs anyway:

Solutions:

- **translate** functional programs to TRSs, while retaining some desirable properties

```
let rec length l =
    match l with
        | [] -> 0
        | _ ::  t -> 1 + length t;;
```

$$\begin{array}{rcl} \mathtt{length}(\mathtt{nil}) & \rightarrow & \mathtt{0} \\ \mathtt{length}(\mathtt{cons}(x,t)) & \rightarrow & \mathtt{s}(\mathtt{length}(t)) \end{array}$$

- **transpose** methods from term rewriting to functional programs
- define **extensions** of basic TRSs

Analysing functional programs anyway:

Solutions:

- **translate** functional programs to TRSs, while retaining some desirable properties

```
let rec length l =
    match l with
        | [] -> 0
        | _ ::  t -> 1 + length t;;
```

$$\begin{array}{rcl} \mathtt{length}(\mathtt{nil}) & \to & \mathtt{0} \\ \mathtt{length}(\mathtt{cons}(x,t)) & \to & \mathtt{length}(t) + 1 \end{array}$$

- **transpose** methods from term rewriting to functional programs

- define **extensions** of basic TRSs

Quiz

1. In ordered resolution, we require that $L \not\succ P$. Why do we not just require $P \succeq L$?
2. What is a Horn clause?
3. Why is it more efficient to use resolution with Horn clauses than in general?
4. Why do we need equational logic when we already have predicate logic?
5. Give the positive superposition and negative superposition rule, and an example of how they might be applied (just one step; no need to give a full superposition proof).