

# Automated Reasoning

Georg Moser

Institute of Computer Science @ UIBK

Winter 2013



# Summary Last Lecture

## Definition

- let  $A$  be closed and rectified
- we define the mapping **rsk** as follows:

$$\text{rsk}(A) = \begin{cases} A & \text{no existential quant. in } A \\ \text{rsk}(A_{-\exists y})\{y \mapsto f(x_1, \dots, x_n)\} & \forall x_1, \dots, \forall x_n <_A \exists y \end{cases}$$

- 1  $\exists y$  is the **first** existential quantifier in  $A$
  - 2  $A_{-\exists y}$  denotes  $A$  after omission of  $\exists y$
  - 3 the Skolem function symbol  $f$  is fresh
- the formula **rsk**( $A$ ) is the **(refutational) structural Skolem form** of  $A$

## Theorem

- 1  $\exists$  a set of sentences  $\mathcal{D}_n$  with  $\text{HC}(\mathcal{D}'_n) = 2^{2^{O(n)}}$  for the structural Skolem form  $\mathcal{D}'_n$
- 2  $\text{HC}(\mathcal{D}''_n) \geq \frac{1}{2}2_n$  for the prenex Skolem form

## Definition (Optimised Skolemisation)

- let  $A$  be a sentence in NNF and  $B = \exists x_1 \cdots x_k (E \wedge F)$  a subformula of  $A$  with  $\mathcal{FVar}(\exists \vec{x}(E \wedge F)) = \{y_1, \dots, y_n\}$
- suppose  $A = C[B]$
- suppose  $A \rightarrow \forall y_1, \dots, y_n \exists x_1 \cdots x_k E$  is valid
- we define an **optimised Skolemisation step** as follows

$$\text{opt\_step}(A) = \forall \vec{y} E\{\dots, x_i \mapsto f_i(\vec{y}), \dots\} \wedge C[F\{\dots, x_i \mapsto f_i(\vec{y}), \dots\}]$$

where  $f_1, \dots, f_k$  are new Skolem function symbols

# Outline of the Lecture

## Early Approaches in Automated Reasoning

short recollection of Herbrand's theorem, Gilmore's prover, method of Davis and Putnam

## Starting Points

resolution, tableau provers, Skolemisation, ordered resolution, redundancy and deletion

## Automated Reasoning with Equality

paramodulation, ordered completion and proof orders, superposition

## Applications of Automated Reasoning

Neuman-Stubblebinde Key Exchange Protocol, group theory, resolution and paramodulation as decision procedure, ...

# Outline of the Lecture

## Early Approaches in Automated Reasoning

short recollection of Herbrand's theorem, Gilmore's prover, method of Davis and Putnam

## Starting Points

resolution, tableau provers, Skolemisation, **ordered resolution**, **redundancy and deletion**

## Automated Reasoning with Equality

paramodulation, ordered completion and proof orders, superposition

## Applications of Automated Reasoning

Neuman-Stubblebinde Key Exchange Protocol, group theory, resolution and paramodulation as decision procedure, ...

## Definitions

- a **proper order** is a irreflexive and transitive relation



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order
- a proper order  $\succ$  on a set  $A$  is **well-founded** (on  $A$ ) if

$$\neg \exists a_1 \succ a_2 \succ \dots \quad a_i \in A$$



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order
- a proper order  $\succ$  on a set  $A$  is **well-founded** (on  $A$ ) if

$$\neg \exists a_1 \succ a_2 \succ \dots \quad a_i \in A$$

- a **well-founded order** is a well-founded proper order



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order
- a proper order  $\succ$  on a set  $A$  is **well-founded** (on  $A$ ) if

$$\neg \exists a_1 \succ a_2 \succ \dots \quad a_i \in A$$

- a **well-founded order** is a well-founded proper order
- a **linear** (or **total**) order fulfills:  
 $\forall a, b \in A, a \neq b, \text{ either } a \succ b \text{ or } b \succ a$



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order
- a proper order  $\succ$  on a set  $A$  is **well-founded** (on  $A$ ) if

$$\neg \exists a_1 \succ a_2 \succ \dots \quad a_i \in A$$

- a **well-founded order** is a well-founded proper order
- a **linear** (or **total**) order fulfills:  
 $\forall a, b \in A, a \neq b, \text{ either } a \succ b \text{ or } b \succ a$
- a **well-order** is a linear well-founded order



## Definitions

- a **proper order** is a irreflexive and transitive relation
- a **quasi-order** is reflexive and transitive
- a **partial order** is an anti-symmetric quasi-order
- a proper order  $\succ$  on a set  $A$  is **well-founded** (on  $A$ ) if

$$\neg \exists a_1 \succ a_2 \succ \dots \quad a_i \in A$$

- a **well-founded order** is a well-founded proper order
- a **linear** (or **total**) order fulfills:  
 $\forall a, b \in A, a \neq b, \text{ either } a \succ b \text{ or } b \succ a$
- a **well-order** is a linear well-founded order

## Example

$\geq$  on  $\mathbb{N}$  is a partial order; we often write  $(\mathbb{N}, \geq)$  to indicate the domain;  
 $(\mathbb{N}, \geq)$  is not well-founded, but  $(\mathbb{N}, >)$  is a well-order

# Orders on Literals

## Definition

- let  $\succ$  be a well-founded and total order on ground atomic formulas



# Orders on Literals

## Definition

- let  $\succ$  be a well-founded and total order on ground atomic formulas
- extend  $\succ$  to a well-founded proper order  $\succ_L$  total on ground literals such that:
  - 1 if  $A \succ B$ , then  $A \succ_L B$  and  $\neg A \succ_L \neg B$
  - 2  $\neg A \succ_L A$



# Orders on Literals

## Definition

- let  $\succ$  be a well-founded and total order on ground atomic formulas
- extend  $\succ$  to a well-founded proper order  $\succ_L$  total on ground literals such that:
  - 1 if  $A \succ B$ , then  $A \succ_L B$  and  $\neg A \succ_L \neg B$
  - 2  $\neg A \succ_L A$

## Example

- consider a well-founded proper order  $\succ$  on atoms that is total on ground atomic formulas
- identify an atom  $A$  with the multiset  $\{A\}$  and  $\neg A$  with  $\{A, A\}$
- set  $\succ_L = \succ^{\text{mul}}$
- $\succ_L$  fulfills the above conditions

# Ordered Resolution Calculus

## Definition

$\sigma$  is ground if  $E\sigma$  is ground

- a literal  $L$  is **maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  
 $M\sigma \succ_L L\sigma$



# Ordered Resolution Calculus

## Definition

$\sigma$  is ground if  $E\sigma$  is ground

- a literal  $L$  is **maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succ_L L\sigma$
- $L$  is **strictly maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succcurlyeq_L L\sigma$ ; here  $\succcurlyeq_L$  denotes the reflexive closure



# Ordered Resolution Calculus

## Definition

$\sigma$  is ground if  $E\sigma$  is ground

- a literal  $L$  is **maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succ_L L\sigma$
- $L$  is **strictly maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succcurlyeq_L L\sigma$ ; here  $\succcurlyeq_L$  denotes the reflexive closure

## Definition

**ordered resolution**

$$\frac{C \vee A \quad D \vee \neg B}{(C \vee D)\sigma}$$

**ordered factoring**

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

- 1  $\sigma$  is a mgu of the atomic formulas  $A$  and  $B$

# Ordered Resolution Calculus

## Definition

$\sigma$  is ground if  $E\sigma$  is ground

- a literal  $L$  is **maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succ_L L\sigma$
- $L$  is **strictly maximal** if  $\exists$  ground  $\sigma$  such that for no other literal  $M$ :  $M\sigma \succcurlyeq_L L\sigma$ ; here  $\succcurlyeq_L$  denotes the reflexive closure

## Definition

ordered resolution

$$\frac{C \vee A \quad D \vee \neg B}{(C \vee D)\sigma}$$

ordered factoring

$$\frac{C \vee A \vee B}{(C \vee A)\sigma}$$

- 1  $\sigma$  is a mgu of the atomic formulas  $A$  and  $B$
- 2  $A\sigma$  is **strictly maximal** with respect to  $C\sigma$ ;  $\neg B\sigma$  is **maximal** with respect to  $D\sigma$

## Example

consider the clause set (constants  $a, b$ , predicates  $P, Q, R, S$ )

$$P(x) \vee Q(x) \vee R(x, y) \quad \neg P(x) \quad \neg Q(a)$$

$$S(a, y) \vee \neg R(a, y) \vee S(x, b) \quad \neg S(a, b) \vee \neg R(a, b)$$

together with the atom order  $P(t_1) \succ Q(t_2) \succ S(t_3, t_4) \succ R(t_5, t_6)$

# Example

consider the clause set (constants  $a, b$ , predicates  $P, Q, R, S$ )

$$P(x) \vee Q(x) \vee R(x, y) \quad \neg P(x) \quad \neg Q(a)$$

$$S(a, y) \vee \neg R(a, y) \vee S(x, b) \quad \neg S(a, b) \vee \neg R(a, b)$$

together with the atom order  $P(t_1) \succ Q(t_2) \succ S(t_3, t_4) \succ R(t_5, t_6)$

$$\Pi \quad \frac{\frac{P(x) \vee Q(x) \vee R(x, y) \quad \neg P(x)}{Q(x) \vee R(x, y)} \quad \neg Q(a)}{R(a, y)} \quad \sigma = \{x \mapsto a\}$$

$$\begin{array}{c} \frac{S(a, y) \vee \neg R(a, y) \vee S(x, b)}{S(a, b) \vee \neg R(a, b)} \sigma_1 \quad \neg S(a, b) \vee \neg R(a, b) \\ \hline \neg R(a, b) \vee \neg R(a, b) \\ \hline \neg R(a, b) \quad \sigma_2 \\ \hline \square \end{array}$$

$\Pi$   
 $R(a, y)$

## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$



## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_{\text{OR}}^n (\text{Res}_{\text{OR}}^*)$  of the operator  $\text{Res}_{\text{OR}}$  is defined as for unrestricted resolution



## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_{\text{OR}}^n (\text{Res}_{\text{OR}}^*)$  of the operator  $\text{Res}_{\text{OR}}$  is defined as for unrestricted resolution

## Theorem

*ordered resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\Box \in \text{Res}_{\text{OR}}^*(\mathcal{C})$*



## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

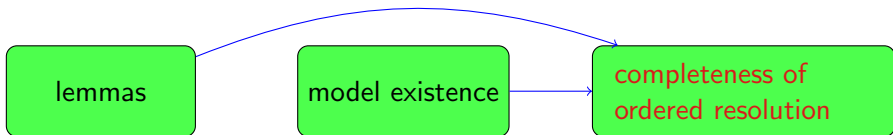
$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_{\text{OR}}^n(\text{Res}_{\text{OR}}^*)$  of the operator  $\text{Res}_{\text{OR}}$  is defined as for unrestricted resolution

## Theorem

*ordered resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\Box \in \text{Res}_{\text{OR}}^*(\mathcal{C})$*

## Proof Plan.



## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_{\text{OR}}^n(\text{Res}_{\text{OR}}^*)$  of the operator  $\text{Res}_{\text{OR}}$  is defined as for unrestricted resolution

## Theorem

*ordered resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\square \in \text{Res}_{\text{OR}}^*(\mathcal{C})$*

## Proof P

$\mathcal{C}$  set of consistent ground clauses  
 $\Rightarrow \mathcal{C}$  admits satisfaction properties  
 + lifting lemmas

lemmas

model existence

completeness of  
ordered resolution

## Definition

- define the **ordered resolution operator**  $\text{Res}_{\text{OR}}(\mathcal{C})$  as follows:

$$\text{Res}_{\text{OR}}(\mathcal{C}) = \{D \mid D \text{ is ordered res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_{\text{OR}}^n(\text{Res}_{\text{OR}}^*)$  of the operator  $\text{Res}_{\text{OR}}$  is defined as for unrestricted resolution

## Theorem

*ordered resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\square \in \text{Res}_{\text{OR}}^*(\mathcal{C})$*

## Proof P

$\mathcal{C}$  set of consistent ground clauses  
 $\Rightarrow \mathcal{C}$  admits satisfaction properties  
 + lifting lemmas

lemmas

model existence

completeness of  
ordered resolution

## Recall

- let  $\mathcal{G}$  be a set of **universal** sentences (of  $\mathcal{L}$ ) **without**  $=$
- $\mathcal{G}$  has a Herbrand model or  $\mathcal{G}$  is unsatisfiable; in the latter case the following statements hold (and are equivalent):
  - 1  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; conjunction  $\bigwedge S$  is unsatisfiable
  - 2  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; disjunction  $\bigvee \{\neg A \mid A \in S\}$  is valid



## Recall

- let  $\mathcal{G}$  be a set of **universal** sentences (of  $\mathcal{L}$ ) **without**  $=$
- $\mathcal{G}$  has a Herbrand model or  $\mathcal{G}$  is unsatisfiable; in the latter case the following statements hold (and are equivalent):
  - 1  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; conjunction  $\bigwedge S$  is unsatisfiable
  - 2  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; disjunction  $\bigvee \{\neg A \mid A \in S\}$  is valid

## Proof of Completeness.

- 1 extend  $\succ_L$  to an order on clauses  $\succ_C$

## Recall

- let  $\mathcal{G}$  be a set of **universal** sentences (of  $\mathcal{L}$ ) **without**  $=$
- $\mathcal{G}$  has a Herbrand model or  $\mathcal{G}$  is unsatisfiable; in the latter case the following statements hold (and are equivalent):
  - 1  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; conjunction  $\bigwedge S$  is unsatisfiable
  - 2  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; disjunction  $\bigvee \{\neg A \mid A \in S\}$  is valid

## Proof of Completeness.

- 1 extend  $\succ_L$  to an order on clauses  $\succ_C$
- 2 a clause set  $\mathcal{C}$  is **maximal** if

$$\neg \exists \mathcal{D} = \mathcal{D}' \cup \{D\} \left( \mathcal{C} = \mathcal{D}' \cup \{D_1, \dots, D_n\}, \forall i \ D \succ_C D_i \right. \\ \left. \text{and there is no } E \in \mathcal{D}', E \succ_C D \right)$$

## Recall

- let  $\mathcal{G}$  be a set of **universal** sentences (of  $\mathcal{L}$ ) **without**  $=$
- $\mathcal{G}$  has a Herbrand model or  $\mathcal{G}$  is unsatisfiable; in the latter case the following statements hold (and are equivalent):
  - 1  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; conjunction  $\bigwedge S$  is unsatisfiable
  - 2  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; disjunction  $\bigvee \{\neg A \mid A \in S\}$  is valid

## Proof of Completeness.

- 1 extend  $\succ_L$  to an order on clauses  $\succ_C$
- 2 a clause set  $\mathcal{C}$  is **maximal** if

$$\neg \exists \mathcal{D} = \mathcal{D}' \cup \{D\} \left( \mathcal{C} = \mathcal{D}' \cup \{D_1, \dots, D_n\}, \forall i \ D \succ_C D_i \right. \\ \left. \text{and there is no } E \in \mathcal{D}', E \succ_C D \right)$$

- 3 choose a maximal unsatisfiable clause set  $\mathcal{C}$   
continue according to proof plan

## Recall

- let  $\mathcal{G}$  be a set of **universal** sentences (of  $\mathcal{L}$ ) **without** =
- $\mathcal{G}$  has a Herbrand model or  $\mathcal{G}$  is unsatisfiable; in the latter case the following statements hold (and are equivalent):
  - 1  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; conjunction  $\bigwedge S$  is unsatisfiable
  - 2  $\exists$  finite subset  $S \subseteq \text{Gr}(\mathcal{G})$ ; disjunction  $\bigvee \{\neg A \mid A \in S\}$  is valid

## Proof of Completeness.

- 1 extend  $\succ_L$  to an order on clauses  $\succ_C$
- 2 a clause set  $\mathcal{C}$  is **maximal** if

$$\neg \exists \mathcal{D} = \mathcal{D}' \cup \{D\} \left( \mathcal{C} = \mathcal{D}' \cup \{D_1, \dots, D_n\}, \forall i \ D \succ_C D_i \right. \\ \left. \text{and there is no } E \in \mathcal{D}', E \succ_C D \right)$$

- 3 choose a maximal unsatisfiable clause set  $\mathcal{C}$   
continue according to proof plan

this proves ground completeness; completeness follows by reformulation of the lifting lemmas

# Lock Resolution

## Definition

a pair  $(L, i)$ ,  $L$  a literal,  $i \in \mathbb{N}$  is a **indexed literal**; different literals are indexed with different numbers



# Lock Resolution

## Definition

a pair  $(L, i)$ ,  $L$  a literal,  $i \in \mathbb{N}$  is a **indexed literal**; different literals are indexed with different numbers

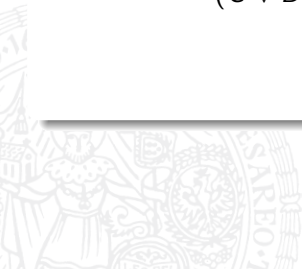
## Definition

**lock resolution**

$$\frac{C \vee (A, i) \quad D \vee (\neg B, j)}{(C \vee D)\sigma}$$

**lock factoring**

$$\frac{C \vee (A, i) \vee (B, j')}{(C \vee (A, i))\sigma}$$



# Lock Resolution

## Definition

a pair  $(L, i)$ ,  $L$  a literal,  $i \in \mathbb{N}$  is a **indexed literal**; different literals are indexed with different numbers

## Definition

**lock resolution**

$$\frac{C \vee (A, i) \quad D \vee (\neg B, j)}{(C \vee D)\sigma}$$

**lock factoring**

$$\frac{C \vee (A, i) \vee (B, j')}{(C \vee (A, i))\sigma}$$

**1**  $\sigma$  is a mgu of the atomic formulas  $A$  and  $B$

# Lock Resolution

## Definition

a pair  $(L, i)$ ,  $L$  a literal,  $i \in \mathbb{N}$  is a **indexed literal**; different literals are indexed with different numbers

## Definition

**lock resolution**

$$\frac{C \vee (A, i) \quad D \vee (\neg B, j)}{(C \vee D)\sigma}$$

**lock factoring**

$$\frac{C \vee (A, i) \vee (B, j')}{(C \vee (A, i))\sigma}$$

- 1  $\sigma$  is a mgu of the atomic formulas  $A$  and  $B$
- 2  $i$  is **minimal** with respect to  $C$ ;  $j$  is **minimal** with respect to  $D$

# Lock Resolution

## Definition

a pair  $(L, i)$ ,  $L$  a literal,  $i \in \mathbb{N}$  is a **indexed literal**; different literals are indexed with different numbers

## Definition

**lock resolution**

$$\frac{C \vee (A, i) \quad D \vee (\neg B, j)}{(C \vee D)\sigma}$$

**lock factoring**

$$\frac{C \vee (A, i) \vee (B, j')}{(C \vee (A, i))\sigma}$$

- 1  $\sigma$  is a mgu of the atomic formulas  $A$  and  $B$
- 2  $i$  is **minimal** with respect to  $C$ ;  $j$  is **minimal** with respect to  $D$

## Remark

indexing represents an a priori literal order, blind on substitutions

# Example

consider the indexed clause set  $\mathcal{C} = \{\neg P^1(x), \neg Q^3(a), \neg S^5(a, b) \vee \neg R^8(a, b), P^2(x) \vee Q^4(x) \vee R^{10}(x, y), S^6(a, y) \vee \neg R^9(a, y) \vee S^7(x, b)\}$

# Example

consider the indexed clause set  $\mathcal{C} = \{\neg P(x), \neg Q(a), \neg S(a, b) \vee \neg R(a, b), P(x) \vee Q(x) \vee R(x, y), S(a, y) \vee \neg R(a, y) \vee S(x, b)\}$

$$\begin{array}{c}
 \frac{\frac{P(x) \vee Q(x) \vee R(x, y) \quad \neg P(x)}{Q(x) \vee R(x, y)} \quad \neg Q(a)}{R(a, y)} \sigma = \{x \mapsto a\} \\
 \Pi \\
 \frac{S(a, y) \vee \neg R(a, y) \vee S(x, b)}{S(a, b) \vee \neg R(a, b)} \sigma_1 \\
 \frac{S(a, b) \vee \neg R(a, b) \quad \neg S(a, b) \vee \neg R(a, b)}{\neg R(a, b) \vee \neg R(a, b)} \\
 \frac{\neg R(a, b) \vee \neg R(a, b)}{\neg R(a, b)} \sigma_2 \\
 \frac{\Pi \quad R(a, y)}{\square}
 \end{array}$$

## Definition

- define the **lock resolution operator**  $\text{Res}_L(\mathcal{C})$  as follows:

$$\text{Res}_L(\mathcal{C}) = \{D \mid D \text{ is lock res./factor with premises in } \mathcal{C}\}$$



## Definition

- define the **lock resolution operator**  $\text{Res}_L(\mathcal{C})$  as follows:

$$\text{Res}_L(\mathcal{C}) = \{D \mid D \text{ is lock res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_L^n$  ( $\text{Res}_L^*$ ) of the operator  $\text{Res}_L$  is defined as for unrestricted resolution



## Definition

- define the **lock resolution operator**  $\text{Res}_L(\mathcal{C})$  as follows:

$$\text{Res}_L(\mathcal{C}) = \{D \mid D \text{ is lock res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_L^n$  ( $\text{Res}_L^*$ ) of the operator  $\text{Res}_L$  is defined as for unrestricted resolution

## Theorem

*lock resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\square \in \text{Res}_L^*(\mathcal{C})$*



## Definition

- define the **lock resolution operator**  $\text{Res}_L(\mathcal{C})$  as follows:

$$\text{Res}_L(\mathcal{C}) = \{D \mid D \text{ is lock res./factor with premises in } \mathcal{C}\}$$

- $n^{\text{th}}$  (unrestricted) iteration  $\text{Res}_L^n$  ( $\text{Res}_L^*$ ) of the operator  $\text{Res}_L$  is defined as for unrestricted resolution

## Theorem

*lock resolution is sound and complete; let  $F$  be a sentence and  $\mathcal{C}$  its clause form; then  $F$  is unsatisfiable iff  $\square \in \text{Res}_L^*(\mathcal{C})$*

## Proof.

lock resolution is a refinement, thus soundness is trivial; completeness follows as for ordered resolution

# Redundancy and Deletion

## Definition

define resolution operator  $\text{Res}(\mathcal{C})$

- $\text{Res}(\mathcal{C}) = \{D \mid D \text{ is resolvent or factor with premises in } \mathcal{C}\}$
- $\text{Res}^0(\mathcal{C}) = \mathcal{C}$ ;  $\text{Res}^{n+1}(\mathcal{C}) := \text{Res}^n(\mathcal{C}) \cup \text{Res}(\text{Res}^n(\mathcal{C}))$
- $\text{Res}^*(\mathcal{C}) := \bigcup_{n \geq 0} \text{Res}^n(\mathcal{C})$



# Redundancy and Deletion

## Definition

define **resolution operator**  $\text{Res}(\mathcal{C})$

- $\text{Res}(\mathcal{C}) = \{D \mid D \text{ is resolvent or factor with premises in } \mathcal{C}\}$
- $\text{Res}^0(\mathcal{C}) = \mathcal{C}$ ;  $\text{Res}^{n+1}(\mathcal{C}) := \text{Res}^n(\mathcal{C}) \cup \text{Res}(\text{Res}^n(\mathcal{C}))$
- $\text{Res}^*(\mathcal{C}) := \bigcup_{n \geq 0} \text{Res}^n(\mathcal{C})$

## Definition

- let  $d(\mathcal{C}) = \min\{n \mid \square \in \text{Res}^n(\mathcal{C})\}$
- the **search complexity** of  $\text{Res}$  wrt clause set  $\mathcal{C}$  is  
 $\text{scomp}(\mathcal{C}) = |\text{Res}^{d(\mathcal{C})}(\mathcal{C})|$

# Redundancy and Deletion

## Definition

define **resolution operator**  $\text{Res}(\mathcal{C})$

- $\text{Res}(\mathcal{C}) = \{D \mid D \text{ is resolvent or factor with premises in } \mathcal{C}\}$
- $\text{Res}^0(\mathcal{C}) = \mathcal{C}$ ;  $\text{Res}^{n+1}(\mathcal{C}) := \text{Res}^n(\mathcal{C}) \cup \text{Res}(\text{Res}^n(\mathcal{C}))$
- $\text{Res}^*(\mathcal{C}) := \bigcup_{n \geq 0} \text{Res}^n(\mathcal{C})$

## Definition

- let  $d(\mathcal{C}) = \min\{n \mid \square \in \text{Res}^n(\mathcal{C})\}$
- the **search complexity** of  $\text{Res}$  wrt clause set  $\mathcal{C}$  is  
 $\text{scomp}(\mathcal{C}) = |\text{Res}^{d(\mathcal{C})}(\mathcal{C})|$

## Question

howto reduce the search complexity (of resolution refinements)?

## Answer

three answers:

### 1 refinements

consider refutational complete **restrictions** of resolution



## Answer

three answers:

**1 refinements**

consider refutational complete **restrictions** of resolution

**2 redundancy tests**

redundancy can appear in the form of circular derivations or in that of tautology clauses



## Answer

three answers:

**1** refinements

consider refutational complete **restrictions** of resolution

**2** redundancy tests

redundancy can appear in the form of circular derivations or in that of tautology clauses

**3** heuristics

...



## Answer

three answers:

**1** refinements

consider refutational complete **restrictions** of resolution

**2** redundancy tests

redundancy can appear in the form of circular derivations or in that of tautology clauses

**3** heuristics

...

## Remarks

- refinements reduce the search space as fewer derivations are possible, however the minimal proof length may be increased

## Answer

three answers:

### 1 refinements

consider refutational complete **restrictions** of resolution

### 2 redundancy tests

redundancy can appear in the form of circular derivations or in that of tautology clauses

### 3 heuristics

...

## Remarks

- refinements reduce the search space as fewer derivations are possible, however the minimal proof length may be increased
- redundancy tests cannot increase the proof length, but may be costly  
call a clause  $D$  **redundant** in  $\mathcal{C}$  if  $\exists C_1, \dots, C_k$  with  $C_1, \dots, C_k \models D$

## Lemma

*application of subsumption and tautology elimination as pre-processing steps preserves completeness*



## Lemma

*application of subsumption and tautology elimination as pre-processing steps preserves completeness*

## Definition

subsumption and resolution can be combined in the following ways

- 1 **forward subsumption**

newly derived clauses subsumed by existing clauses are deleted

## Lemma

*application of subsumption and tautology elimination as pre-processing steps preserves completeness*

## Definition

subsumption and resolution can be combined in the following ways

- 1 forward subsumption  
newly derived clauses subsumed by existing clauses are deleted
- 2 **backward subsumption**  
existing clauses  $C$  subsumed by newly derived clauses  $D$  become inactive  
inactive clauses are reactivated, if  $D$  is no ancestor of current clause

## Lemma

*application of subsumption and tautology elimination as pre-processing steps preserves completeness*

## Definition

subsumption and resolution can be combined in the following ways

- 1 forward subsumption  
newly derived clauses subsumed by existing clauses are deleted
- 2 backward subsumption  
existing clauses  $C$  subsumed by newly derived clauses  $D$  become inactive  
inactive clauses are reactivated, if  $D$  is no ancestor of current clause
- 3 replacement  
the set of all clauses (derived and intital) are frequently reduced under subsumption

## Lemma

*application of subsumption and tautology elimination as pre-processing steps preserves completeness*

## Definition

subsumption and resolution can be combined in the following ways

- 1 forward subsumption  
newly derived clauses subsumed by existing clauses are deleted
- 2 backward subsumption  
existing clauses  $C$  subsumed by newly derived clauses  $D$  become inactive  
inactive clauses are reactivated, if  $D$  is no ancestor of current clause
- 3 replacement  
the set of all clauses (derived and intital) are frequently reduced under subsumption

# Tautology Elimination

## Definition

- a clause  $C$  containing complementary literals is a tautology
- **tautology elimination** is the process of removing newly derived tautological clauses (that is, we assume the initial clause set is taut-reduced)



# Tautology Elimination

## Definition

- a clause  $C$  containing complementary literals is a tautology
- **tautology elimination** is the process of removing newly derived tautological clauses (that is, we assume the initial clause set is taut-reduced)

## Example

consider the clause

$$P(f(a, b)) \vee \neg P(f(x, b)) \vee \neg P(f(a, y))$$

factoring yields the tautology  $P(f(a, b)) \vee \neg P(f(a, b))$

## Example

consider the following (tautology free) clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

we have  $\text{scomp}(\mathcal{C}) = 15$  for unrestricted resolution; however the following resolution steps derive tautologies

$$\frac{P(x) \vee R(x) \quad \neg P(x) \vee \neg R(x)}{P(x) \vee \neg P(x)}$$

$$\frac{P(x) \vee R(x) \quad \neg P(x) \vee \neg R(x)}{R(x) \vee \neg R(x)}$$



## Example

consider the following (tautology free) clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

we have  $\text{scomp}(\mathcal{C}) = 15$  for unrestricted resolution; however the following resolution steps derive tautologies

$$\frac{P(x) \vee R(x) \quad \neg P(x) \vee \neg R(x)}{P(x) \vee \neg P(x)}$$

$$\frac{P(x) \vee R(x) \quad \neg P(x) \vee \neg R(x)}{R(x) \vee \neg R(x)}$$

## Lemma

- 1 *tautology elimination is incomplete for lock resolution*
- 2 *tautology elimination is complete for unrestricted and ordered resolution*

## Theorem

- 1 *(ordered) resolution (for any admissible atom order) is complete under forward subsumption*
- 2 *forward subsumption does not increase the search complexity of (ordered) resolution*



## Theorem

- 1 *(ordered) resolution (for any admissible atom order) is complete under forward subsumption*
- 2 *forward subsumption does not increase the search complexity of (ordered) resolution*

## Proof Sketch.

- 1 let  $C'$ ,  $C$ ,  $D'$ ,  $D$  be clauses such that  $C'$  subsumes  $C$  and  $D'$  subsumes  $D$

## Theorem

- 1 *(ordered) resolution (for any admissible atom order) is complete under forward subsumption*
- 2 *forward subsumption does not increase the search complexity of (ordered) resolution*

## Proof Sketch.

- 1 let  $C'$ ,  $C$ ,  $D'$ ,  $D$  be clauses such that  $C'$  subsumes  $C$  and  $D'$  subsumes  $D$
- 2 one shows that if  $E$  is a resolvent of  $C$  and  $D$ , then one of the following cases happens:
  - $C'$  subsumes  $E$
  - $D'$  subsumes  $E$
  - $\exists$  resolvent  $E'$  of  $C'$  and  $D'$  such that  $E'$  subsumes  $E$

## Theorem

- 1 *(ordered) resolution (for any admissible atom order) is complete under forward subsumption*
- 2 *forward subsumption does not increase the search complexity of (ordered) resolution*

## Proof Sketch.

- 1 let  $C'$ ,  $C$ ,  $D'$ ,  $D$  be clauses such that  $C'$  subsumes  $C$  and  $D'$  subsumes  $D$
- 2 one shows that if  $E$  is a resolvent of  $C$  and  $D$ , then one of the following cases happens:
  - $C'$  subsumes  $E$
  - $D'$  subsumes  $E$
  - $\exists$  resolvent  $E'$  of  $C'$  and  $D'$  such that  $E'$  subsumes  $E$
- 3 using this observation in an inductive argument, completeness follows

## Theorem

- 1 *(ordered) resolution (for any admissible atom order) is complete under forward subsumption*
- 2 *forward subsumption does not increase the search complexity of (ordered) resolution*

## Proof Sketch.

- 1 let  $C'$ ,  $C$ ,  $D'$ ,  $D$  be clauses such that  $C'$  subsumes  $C$  and  $D'$  subsumes  $D$
- 2 one shows that if  $E$  is a resolvent of  $C$  and  $D$ , then one of the following cases happens:
  - $C'$  subsumes  $E$
  - $D'$  subsumes  $E$
  - $\exists$  resolvent  $E'$  of  $C'$  and  $D'$  such that  $E'$  subsumes  $E$
- 3 using this observation in an inductive argument, completeness follows

## Lemma

*lock resolution is not complete under forward subsumption*



## Lemma

*lock resolution is not complete under forward subsumption*

## Proof.

- 1 let  $C, D$  be indexed clauses; we say an  $C$  **subsumes**  $D$  if the the clause part of  $C$  subsumes the clause part of  $D$

## Lemma

*lock resolution is not complete under forward subsumption*

## Proof.

- 1 let  $C, D$  be indexed clauses; we say an  $C$  **subsumes**  $D$  if the the clause part of  $C$  subsumes the clause part of  $D$
- 2 consider the following clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

## Lemma

*lock resolution is not complete under forward subsumption*

## Proof.

- 1 let  $C, D$  be indexed clauses; we say an  $C$  **subsumes**  $D$  if the the clause part of  $C$  subsumes the clause part of  $D$
- 2 consider the following clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

- 3 the following clauses are derivable by lock resolution and essential to derive  $\square$

$$R(x) \vee \neg P(x) \quad \neg P(x) \vee \neg R(x)$$

## Lemma

*lock resolution is not complete under forward subsumption*

## Proof.

- 1 let  $C, D$  be indexed clauses; we say an  $C$  **subsumes**  $D$  if the clause part of  $C$  subsumes the clause part of  $D$
- 2 consider the following clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

- 3 the following clauses are derivable by lock resolution and essential to derive  $\square$

$$R(x) \vee \neg P(x) \quad \neg P(x) \vee \neg R(x)$$

- 4 however these are subsumed by  $R(x) \vee \neg P(x)$  and  $\neg P(x) \vee \neg R(x)$  respectively

## Lemma

*lock resolution is not complete under forward subsumption*

## Proof.

- 1 let  $C, D$  be indexed clauses; we say an  $C$  **subsumes**  $D$  if the clause part of  $C$  subsumes the clause part of  $D$
- 2 consider the following clause set  $\mathcal{C}$

$$P(x) \vee R(x) \quad R(x) \vee \neg P(x) \quad P(x) \vee \neg R(x) \quad \neg P(x) \vee \neg R(x)$$

- 3 the following clauses are derivable by lock resolution and essential to derive  $\square$

$$R(x) \vee \neg P(x) \quad \neg P(x) \vee \neg R(x)$$

- 4 however these are subsumed by  $R(x) \vee \neg P(x)$  and  $\neg P(x) \vee \neg R(x)$  respectively

## Example

consider the following set of clauses

$$C_1: P(f(x)) \vee R(x) \vee \neg P(f(x))$$

$$C_2: P(x) \vee Q(x)$$

$$C_3: R(f(x))$$

$$C_4: Q(x) \vee \neg R(x)$$

$$C_5: \neg Q(f(x))$$

$C_1$  can be resolved with  $C_2$ ,  $C_4$  and itself



## Example

consider the following set of clauses

$$C_1: P(f(x)) \vee R(x) \vee \neg P(f(x))$$

$$C_2: P(x) \vee Q(x)$$

$$C_3: R(f(x))$$

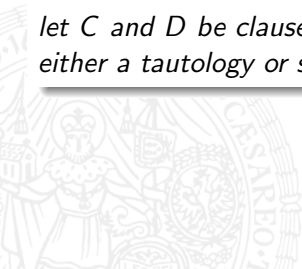
$$C_4: Q(x) \vee \neg R(x)$$

$$C_5: \neg Q(f(x))$$

$C_1$  can be resolved with  $C_2$ ,  $C_4$  and itself

## Lemma

*let  $C$  and  $D$  be clauses and  $C$  a tautology; any resolvent of  $C$  and  $D$  is either a tautology or subsumed by  $D$*



## Example

consider the following set of clauses

$$C_1: P(f(x)) \vee R(x) \vee \neg P(f(x))$$

$$C_2: P(x) \vee Q(x)$$

$$C_3: R(f(x))$$

$$C_4: Q(x) \vee \neg R(x)$$

$$C_5: \neg Q(f(x))$$

$C_1$  can be resolved with  $C_2$ ,  $C_4$  and itself

## Lemma

*let  $C$  and  $D$  be clauses and  $C$  a tautology; any resolvent of  $C$  and  $D$  is either a tautology or subsumed by  $D$*

## Theorem

*(ordered) resolution is complete under forward subsumption and tautology elimination*