

First-Order Logic: Theory and Practice

Christoph Benzmüller

Freie Universität Berlin

Block Lecture, WS 2012, October 1-12, 2012

First-Order Logic — Mechanization/Automation

(this part of the lecture very closely follows Fitting's textbook)

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language L
 - ▶ The proofs we will construct will use sentences of L^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language L
 - ▶ The proofs we will construct will use sentences of L^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language L
 - ▶ The proofs we will construct will use sentences of L^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language \mathcal{L}
 - ▶ The proofs we will construct will use sentences of \mathcal{L}^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language \mathcal{L}
 - ▶ The proofs we will construct will use sentences of \mathcal{L}^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language \mathcal{L}
 - ▶ The proofs we will construct will use sentences of \mathcal{L}^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language \mathbf{L}
 - ▶ The proofs we will construct will use sentences of \mathbf{L}^{par}

- ▶ We will again study two methods: tableaux and resolution
 - ▶ first version(s): simple to explain
 - ▶ second version(s): better suited for automation
- ▶ We will address natural deduction and sequent calculi later in this course
- ▶ New constant symbols (parameters) have been routinely introduced the previous part; they also appeared in the first-order consistency property
- ▶ Not very surprisingly, they will play an important role also in the proof procedures below:
 - ▶ The statements we want to prove will be sentences of a language \mathbf{L}
 - ▶ The proofs we will construct will use sentences of \mathbf{L}^{par}

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{par}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all propositional Tableau Expansion Rules

- ▶ Tableau proofs are closed trees (as seen before)
 - ▶ composed of the rules we already know from propositional logic
 - ▶ but with two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in a branch of a tableau) — parameters

Definition — First-Order Tableau Expansion Rules

159

$\frac{\gamma}{\gamma(t)}$ for any closed term t of L^{par}

$\frac{\delta}{\delta(p)}$ for a new parameter p

plus all propositional Tableau Expansion Rules

Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

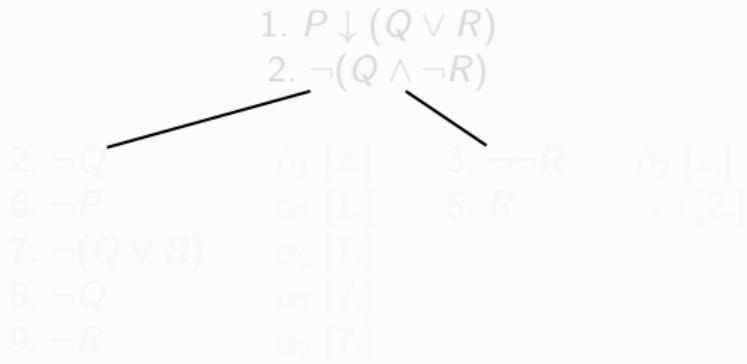
$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160



Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160

1.	$P \downarrow (Q \vee R)$			
2.	$\neg(Q \wedge \neg R)$			
2.	$\neg Q$	$\beta_1 [2.]$	3. $\neg\neg R$	$\beta_2 [2.]$
6.	$\neg P$	$\alpha_1 [1.]$	5. R	$\neg\neg [2.]$
7.	$\neg(Q \vee R)$	$\alpha_2 [1.]$		
8.	$\neg Q$	$\alpha_1 [7.]$		
9.	$\neg R$	$\alpha_2 [7.]$		

Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

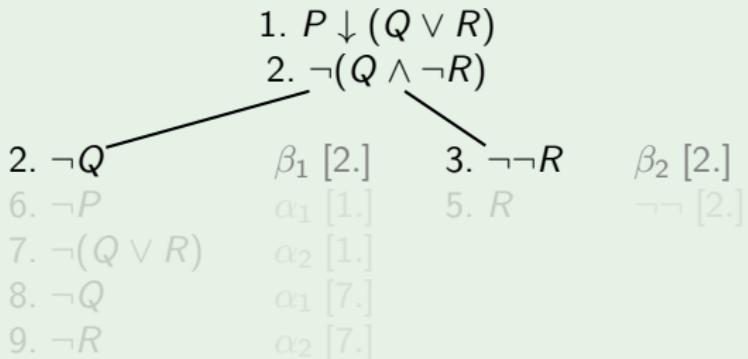
$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160



Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

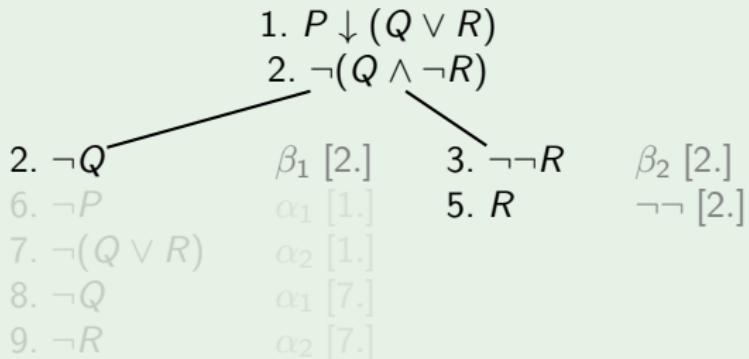
$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160



Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

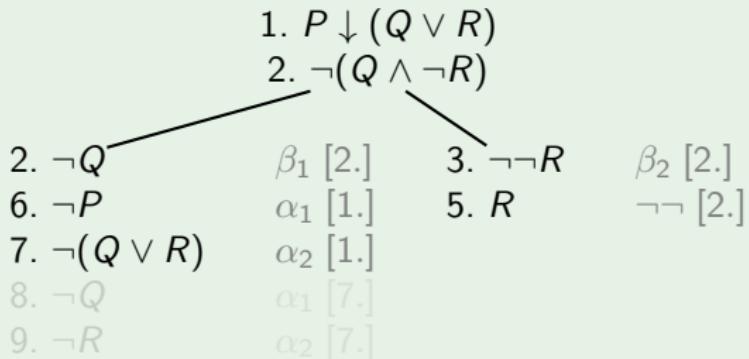
$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160



Remember: Propositional Tableau Expansion Rules

$$\frac{\neg\neg Z}{Z}$$

$$\frac{\neg T}{\perp}$$

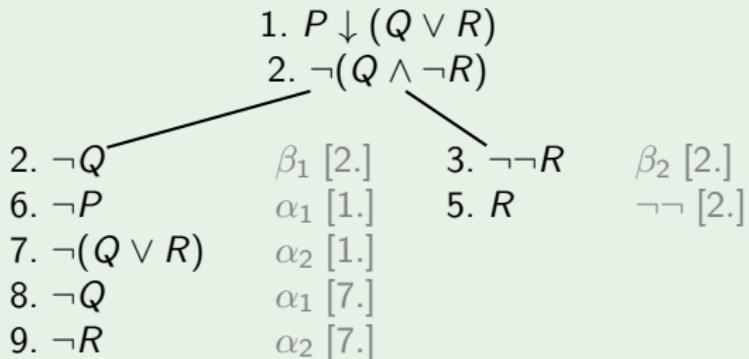
$$\frac{\neg\perp}{T}$$

$$\frac{\alpha_1}{\alpha_2}$$

$$\frac{\beta}{\beta_1 \mid \beta_2}$$

Example — Tableau for $\{P \downarrow (Q \vee R), \neg(Q \wedge \neg R)\}$

160



Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
 2. $(\forall x)(P(x) \vee Q(x))$ α_1 [1.]
 3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ α_2 [1.]
 4. $\neg(\exists x)P(x)$ α_1 [3.]
 5. $\neg(\forall x)Q(x)$ α_2 [3.]
 6. $\neg Q(c)$ $\delta(c)$ [5.]
 7. $\neg P(c)$ $\gamma(c)$ [4.]
 8. $P(c) \vee Q(c)$ $\gamma(c)$ [2.]
-
9. $P(c)$ β_1 [8.]
 10. $Q(c)$ β_2 [8.]

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
 2. $(\forall x)(P(x) \vee Q(x))$ α_1 [1.]
 3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ α_2 [1.]
 4. $\neg(\exists x)P(x)$ α_1 [3.]
 5. $\neg(\forall x)Q(x)$ α_2 [3.]
 6. $\neg Q(c)$ $\delta(c)$ [5.]
 7. $\neg P(c)$ $\gamma(c)$ [4.]
 8. $P(c) \vee Q(c)$ $\gamma(c)$ [2.]
-
9. $P(c)$ β_1 [8.]
 10. $Q(c)$ β_2 [8.]

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
6. $\neg Q(c)$ $\delta(c) [5.]$
7. $\neg P(c)$ $\gamma(c) [4.]$
8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$

9. $P(c)$ $\beta_1 [8.]$
10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
6. $\neg Q(c)$ $\delta(c) [5.]$
7. $\neg P(c)$ $\gamma(c) [4.]$
8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$
9. $P(c)$ $\beta_1 [8.]$
10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
6. $\neg Q(c)$ $\delta(c) [5.]$
7. $\neg P(c)$ $\gamma(c) [4.]$
8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$

9. $P(c)$ $\beta_1 [8.]$
10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
 2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
 3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$
-
9. $P(c)$ $\beta_1 [8.]$
 10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
 2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
 3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$
-
9. $P(c)$ $\beta_1 [8.]$
 10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
2. $(\forall x)(P(x) \vee Q(x))$ α_1 [1.]
3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ α_2 [1.]
4. $\neg(\exists x)P(x)$ α_1 [3.]
5. $\neg(\forall x)Q(x)$ α_2 [3.]
6. $\neg Q(c)$ $\delta(c)$ [5.]
7. $\neg P(c)$ $\gamma(c)$ [4.]
8. $P(c) \vee Q(c)$ $\gamma(c)$ [2.]
9. $P(c)$ β_1 [8.]
10. $Q(c)$ β_2 [8.]

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

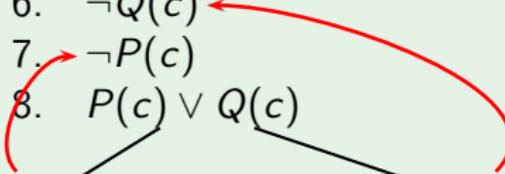
1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
6. $\neg Q(c)$ $\delta(c) [5.]$
7. $\neg P(c)$ $\gamma(c) [4.]$
8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$
9. $P(c)$ $\beta_1 [8.]$
10. $Q(c)$ $\beta_2 [8.]$

Example

161

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we construct a closed tableau for

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $\neg[(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))]$
 2. $(\forall x)(P(x) \vee Q(x))$ $\alpha_1 [1.]$
 3. $\neg((\exists x)P(x) \vee (\forall x)Q(x))$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(c) \vee Q(c)$ $\gamma(c) [2.]$
 9. $P(c)$ $\beta_1 [8.]$
 10. $Q(c)$ $\beta_2 [8.]$
- 

Remarks:

- ▶ if a branch closes because it contains Z and $\neg Z$ we say it *closes on Z*
- ▶ we may close on non-atomic formulas
- ▶ if a closure is possible, then closure at the atomic level is possible
- ▶ tableau rules are non-deterministic: they say what *can* be done and not what *must* be done
- ▶ unlike in propositional case we may now work forever: the problem is the γ -rule (see next slide)

Remarks:

- ▶ if a branch closes because it contains Z and $\neg Z$ we say it *closes on Z*
- ▶ we may close on non-atomic formulas
- ▶ if a closure is possible, then closure at the atomic level is possible
- ▶ tableau rules are non-deterministic: they say what *can* be done and not what *must* be done
- ▶ unlike in propositional case we may now work forever: the problem is the γ -rule (see next slide)

Remarks:

- ▶ if a branch closes because it contains Z and $\neg Z$ we say it *closes on Z*
- ▶ we may close on non-atomic formulas
- ▶ if a closure is possible, then closure at the atomic level is possible
- ▶ tableau rules are non-deterministic: they say what *can* be done and not what *must* be done
- ▶ unlike in propositional case we may now work forever: the problem is the γ -rule (see next slide)

Remarks:

- ▶ if a branch closes because it contains Z and $\neg Z$ we say it *closes on Z*
- ▶ we may close on non-atomic formulas
- ▶ if a closure is possible, then closure at the atomic level is possible
- ▶ tableau rules are non-deterministic: they say what *can* be done and not what *must* be done
- ▶ unlike in propositional case we may now work forever: the problem is the γ -rule (see next slide)

Remarks:

- ▶ if a branch closes because it contains Z and $\neg Z$ we say it *closes on Z*
- ▶ we may close on non-atomic formulas
- ▶ if a closure is possible, then closure at the atomic level is possible
- ▶ tableau rules are non-deterministic: they say what *can* be done and not what *must* be done
- ▶ unlike in propositional case we may now work forever: the problem is the γ -rule (see next slide)

Example

162

In order to prove $(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]$ we construct a closed tableau for its negation:

1. $\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}$
 2. $(\forall x)[P(x) \vee Q(x)]$ $\alpha_1 [1.]$
 3. $\neg[(\exists x)P(x) \vee (\forall x)Q(x)]$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(t_1) \vee Q(t_1)$ $\gamma(t_1) [2.]$
 9. $P(t_2) \vee Q(t_2)$ $\gamma(t_2) [2.]$
 10. $P(t_3) \vee Q(t_3)$ $\gamma(t_3) [2.]$
- ...

Example

162

In order to prove $(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]$ we construct a closed tableau for its negation:

1. $\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}$
 2. $(\forall x)[P(x) \vee Q(x)]$ $\alpha_1 [1.]$
 3. $\neg[(\exists x)P(x) \vee (\forall x)Q(x)]$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(t_1) \vee Q(t_1)$ $\gamma(t_1) [2.]$
 9. $P(t_2) \vee Q(t_2)$ $\gamma(t_2) [2.]$
 10. $P(t_3) \vee Q(t_3)$ $\gamma(t_3) [2.]$
- ...

Example

162

In order to prove $(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]$ we construct a closed tableau for its negation:

1. $\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}$
2. $(\forall x)[P(x) \vee Q(x)]$ $\alpha_1 [1.]$
3. $\neg[(\exists x)P(x) \vee (\forall x)Q(x)]$ $\alpha_2 [1.]$
4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
6. $\neg Q(c)$ $\delta(c) [5.]$
7. $\neg P(c)$ $\gamma(c) [4.]$
8. $P(t_1) \vee Q(t_1)$ $\gamma(t_1) [2.]$
9. $P(t_2) \vee Q(t_2)$ $\gamma(t_2) [2.]$
10. $P(t_3) \vee Q(t_3)$ $\gamma(t_3) [2.]$

...

Example

162

In order to prove $(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]$ we construct a closed tableau for its negation:

1. $\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}$
 2. $(\forall x)[P(x) \vee Q(x)]$ $\alpha_1 [1.]$
 3. $\neg[(\exists x)P(x) \vee (\forall x)Q(x)]$ $\alpha_2 [1.]$
 4. $\neg(\exists x)P(x)$ $\alpha_1 [3.]$
 5. $\neg(\forall x)Q(x)$ $\alpha_2 [3.]$
 6. $\neg Q(c)$ $\delta(c) [5.]$
 7. $\neg P(c)$ $\gamma(c) [4.]$
 8. $P(t_1) \vee Q(t_1)$ $\gamma(t_1) [2.]$
 9. $P(t_2) \vee Q(t_2)$ $\gamma(t_2) [2.]$
 10. $P(t_3) \vee Q(t_3)$ $\gamma(t_3) [2.]$
- ...

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
 - ▶ but what to do with the γ -rule?
 - ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Remarks (cont'd):

- ▶ in this proof attempt it is obvious what to do, but in complex situations it is not
- ▶ in fact, first-order logic does not have a decision procedure
- ▶ all we can do is to organize a proof search in a 'clever' way
- ▶ but generally we may search forever (if a statement is a non-theorem)
- ▶ we could enforce a notion of *strictness*: forbid formula reuse
- ▶ but what to do with the γ -rule?
- ▶ reuse of γ -rule is generally required; see e.g.
 $(\forall x)P(x) \supset [P(a) \wedge P(b)]$
- ▶ need for reuse of γ -rule is related to the lack of a decision procedure for first-order logic
- ▶ for the moment we do not impose any strictness conditions
- ▶ good heuristics are nevertheless useful; give preference as follows: propositional rules, δ -rules, γ -rules

Question: How to prove $S \models X$ problems?

Definition — S -Introduction Rule for Tableau

163

Given a set of axioms S . *Any member of S can be added to the end of any branch.*

We write $S \vdash_{ft} X$ if there is a closed propositional tableau for $\{\neg X\}$ when allowing the S -Introduction Rule for Tableau to be applied with the axioms in S .

Question: How to prove $S \models X$ problems?

Definition — S -Introduction Rule for Tableau

163

Given a set of axioms S . *Any member of S can be added to the end of any branch.*

We write $S \vdash_{ft} X$ if there is a closed propositional tableau for $\{\neg X\}$ when allowing the S -Introduction Rule for Tableau to be applied with the axioms in S .

Example

164

We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$

1. $\neg[P(a) \wedge Q(a)]$
2. $\neg P(a) \quad \beta_1 [1.]$
3. $(\forall x)P(x) \quad \text{S-intro}$
4. $P(a) \quad \delta(a) [2.]$
5. $(\forall y)Q(y) \quad \text{S-intro}$
6. $Q(a) \quad \delta(a) [5.]$
7. $\neg Q(a) \quad \beta_2 [1.]$

Example

164

We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$

1. $\neg[P(a) \wedge Q(a)]$

2. $\neg P(a) \quad \beta_1 [1.]$

3. $(\forall x)P(x) \quad \text{S-intro}$

4. $P(a) \quad \delta(a) [2.]$

7. $\neg Q(a) \quad \beta_2 [1.]$

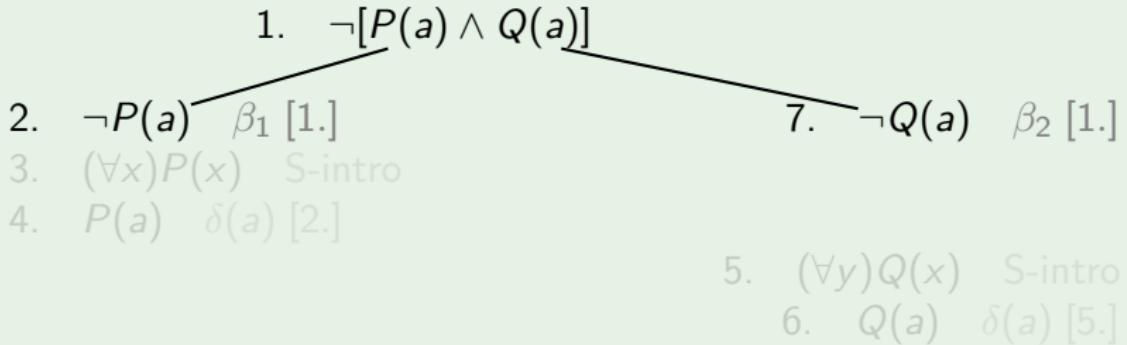
5. $(\forall y)Q(x) \quad \text{S-intro}$

6. $Q(a) \quad \delta(a) [5.]$

Example

164

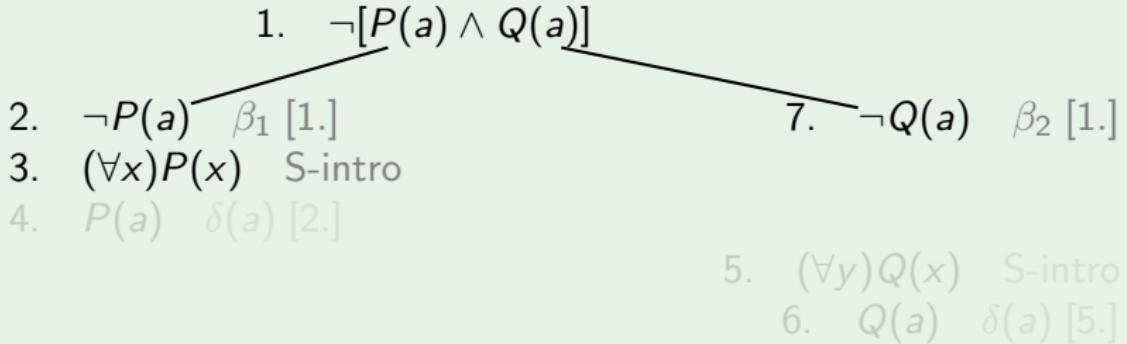
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

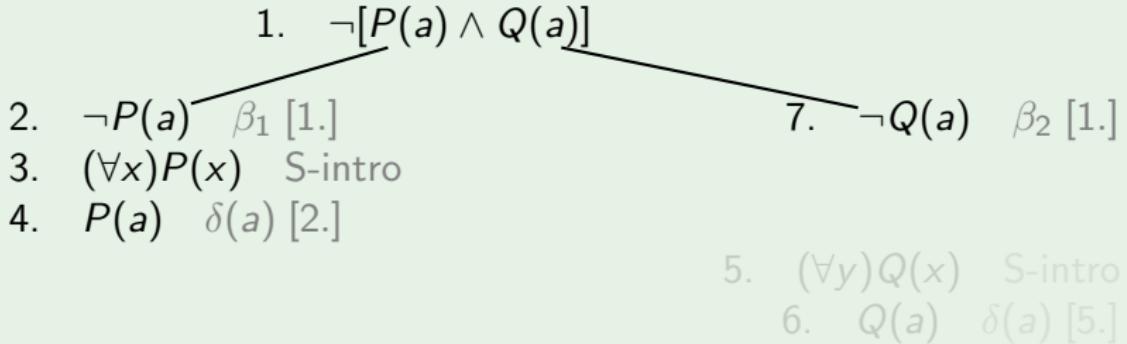
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

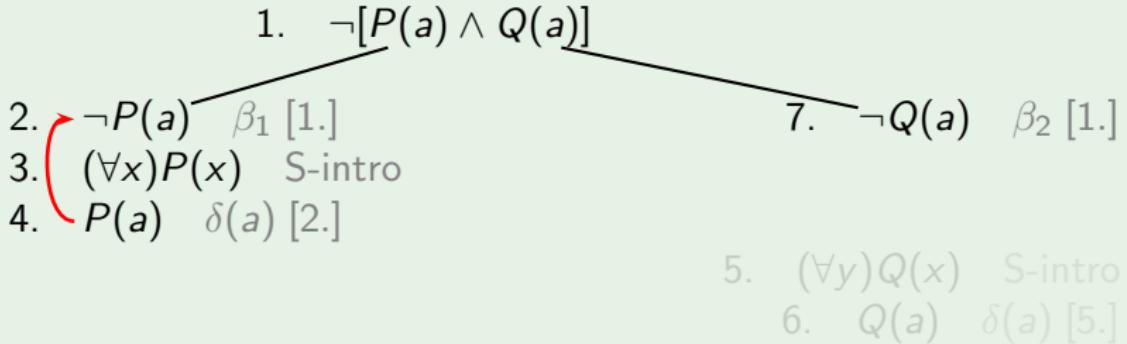
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

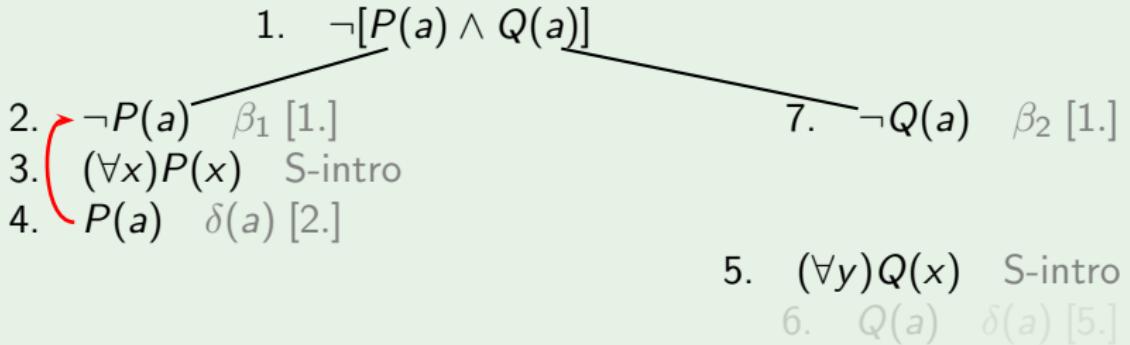
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

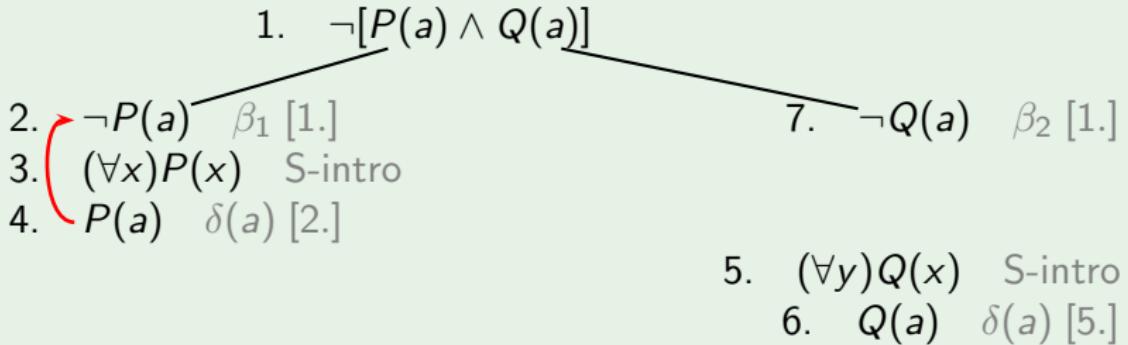
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

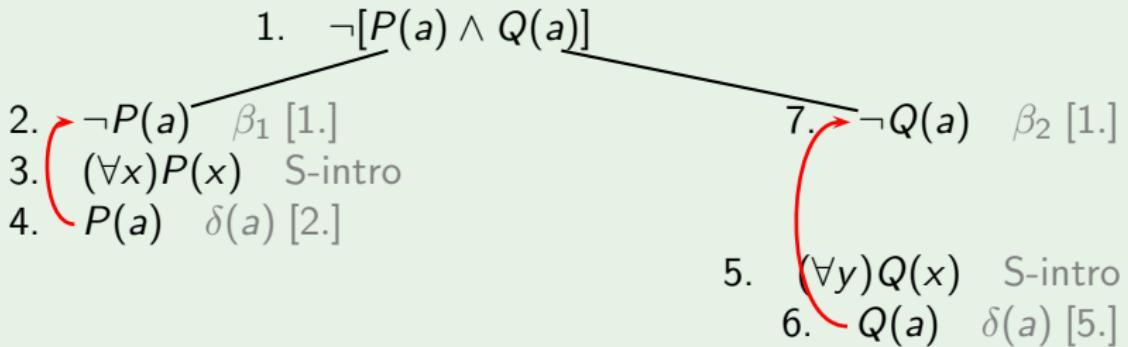
We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Example

164

We prove $\{(\forall x)P(x), (\forall y)Q(y)\} \vdash_{ft} P(a) \wedge Q(a)$



Exercises:

1. Prove resp. try to prove the following (some are non-theorems):

1.1 $(\exists x)(\forall y)R(x, y) \supset (\forall y)(\exists x)R(x, y)$

1.2 $(\forall x)(\exists y)R(x, y) \supset (\exists y)(\forall x)R(x, y)$

1.3 $(\exists x)[P(x) \supset (\forall x)P(x)]$

1.4 $(\exists x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\exists x)Q(x)]$

1.5 $(\exists x)[P(x) \wedge Q(x)] \supset [(\exists x)P(x) \wedge (\exists x)Q(x)]$

1.6 $[(\exists x)P(x) \wedge (\forall x)Q(x)] \supset (\exists x)[P(x) \wedge Q(x)]$

1.7 $(\forall x)(\exists y)(\forall z)(\exists w)[R(x, y) \vee \neg R(w, z)]$

1.8 $(\exists x)(\forall y)[P(y) \uparrow (P(x) \uparrow Q(x))] \supset (\forall x)Q(x)$

1.9 $(\forall x)[P(x) \supset Q] \supset [(\exists x)P(x) \supset Q]$

(where $x \notin \text{FreeVariables-of}(Q)$)

1.10 $(\forall x)[P(x) \supset Q] \supset [(\forall x)P(x) \supset Q]$

(where $x \notin \text{FreeVariables-of}(Q)$)

Exercises (cont'd):

2. We define

- ▶ $\text{trans} := (\forall x)(\forall y)(\forall z)\{[R(x, y) \wedge R(y, z)] \supset R(x, z)\}$
- ▶ $\text{sym} := (\forall x)(\forall y)[R(x, y) \supset R(y, x)]$
- ▶ $\text{ref} := (\forall x)R(x, x)$
- ▶ $\text{nontriv} := (\forall x)(\exists y)R(x, y)$

- (1) Show $\{\text{trans}, \text{sym}\} \not\models_f \text{ref}$ by producing a model in which trans and sym are true but ref is not.
- (2) Show $\{\text{trans}, \text{sym}, \text{nontriv}\} \vdash_{ft} \text{ref}$

3. Prove

- 3.1 $(\exists x)(\forall y)(\forall z)[(P(y) \supset Q(z)) \supset (P(x) \supset Q(x))]$
- 3.2 $(\exists x)(\forall y)(\forall z)[(P(y) \vee Q(z)) \supset (P(x) \vee Q(x))]$
- 3.3 $(\exists x)(\forall y)(\forall z)(\forall w)[(P(y) \vee Q(z) \vee R(w)) \supset (P(x) \vee Q(x) \vee R(x))]$

4. Prove, by structural induction, that if there exists a closed first-order tableau for a set S then there is a closed tableau for S in which all closures are on atomic sentences.

Remember: Resolution works on conjunction of disjunctions

Input: $\langle D_1, \dots, D_n \rangle$ where the D_i are clauses (disjunctions)

$$\frac{\langle D_1, \dots, [\dots, \neg\neg Z, \dots], \dots, D_n \rangle}{\langle D_1, \dots, [\dots, Z, \dots], \dots, D_n \rangle} \quad \frac{\neg\neg Z}{Z}$$

$$\frac{\langle D_1, \dots, [\dots, \neg\top, \dots], \dots, D_n \rangle}{\langle D_1, \dots, [\dots, \perp, \dots], \dots, D_n \rangle} \quad \frac{\neg\top}{\perp}$$

analogous for $\frac{\neg\perp}{\top}$

$$\frac{\langle D_1, \dots, [\dots, \beta, \dots], \dots, D_n \rangle}{\langle D_1, \dots, [\dots, \beta_1, \beta_2, \dots], \dots, D_n \rangle} \quad \frac{\begin{matrix} \beta \\ \beta_1 \\ \beta_2 \end{matrix}}{\beta_1 \mid \beta_2}$$

$$\frac{\langle D_1, \dots, [\dots, \alpha, \dots], \dots, D_n \rangle}{\langle D_1, \dots, [\dots, \alpha_1, \dots], [\dots, \alpha_2, \dots], \dots, D_n \rangle} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

1.	$[P \downarrow (Q \wedge R)]$	
2.	$[\neg(Q \vee (P \supset Q))]$	
3.	$[\neg P]$	$\alpha_1 [1.]$
4.	$[\neg(Q \wedge R)]$	$\alpha_2 [1.]$
5.	$[\neg Q, \neg R]$	$\beta [4.]$
6.	$[\neg Q]$	$\alpha_1 [2.]$
7.	$[\neg(P \supset Q)]$	$\alpha_2 [2.]$
8.	$[\neg\neg P]$	$\alpha_1 [7.]$
9.	$[Q]$	$\alpha_2 [7.]$
10.	$[P]$	$\neg\neg [9.]$

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|-----------------|--|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | $\alpha_1 [1.]$ | |
| 4. | $[\neg(Q \wedge R)]$ | $\alpha_2 [1.]$ | |
| 5. | $[\neg Q, \neg R]$ | $\beta [4.]$ | |
| 6. | $[\neg Q]$ | $\alpha_1 [2.]$ | |
| 7. | $[\neg(P \supset Q)]$ | $\alpha_2 [2.]$ | |
| 8. | $[\neg\neg P]$ | $\alpha_1 [7.]$ | |
| 9. | $[Q]$ | $\alpha_2 [7.]$ | |
| 10. | $[P]$ | $\neg\neg [9.]$ | |

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|-----------------|--|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | $\alpha_1 [1.]$ | |
| 4. | $[\neg(Q \wedge R)]$ | $\alpha_2 [1.]$ | |
| 5. | $[\neg Q, \neg R]$ | $\beta [4.]$ | |
| 6. | $[\neg Q]$ | $\alpha_1 [2.]$ | |
| 7. | $[\neg(P \supset Q)]$ | $\alpha_2 [2.]$ | |
| 8. | $[\neg\neg P]$ | $\alpha_1 [7.]$ | |
| 9. | $[Q]$ | $\alpha_2 [7.]$ | |
| 10. | $[P]$ | $\neg\neg [9.]$ | |

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|-----------------|--|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | $\alpha_1 [1.]$ | |
| 4. | $[\neg(Q \wedge R)]$ | $\alpha_2 [1.]$ | |
| 5. | $[\neg Q, \neg R]$ | $\beta [4.]$ | |
| 6. | $[\neg Q]$ | $\alpha_1 [2.]$ | |
| 7. | $[\neg(P \supset Q)]$ | $\alpha_2 [2.]$ | |
| 8. | $[\neg\neg P]$ | $\alpha_1 [7.]$ | |
| 9. | $[Q]$ | $\alpha_2 [7.]$ | |
| 10. | $[P]$ | $\neg\neg [9.]$ | |

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|------------|------|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | α_1 | [1.] |
| 4. | $[\neg(Q \wedge R)]$ | α_2 | [1.] |
| 5. | $[\neg Q, \neg R]$ | β | [4.] |
| 6. | $[\neg Q]$ | α_1 | [2.] |
| 7. | $[\neg(P \supset Q)]$ | α_2 | [2.] |
| 8. | $[\neg\neg P]$ | α_1 | [7.] |
| 9. | $[Q]$ | α_2 | [7.] |
| 10. | $[P]$ | $\neg\neg$ | [9.] |

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|------------|------|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | α_1 | [1.] |
| 4. | $[\neg(Q \wedge R)]$ | α_2 | [1.] |
| 5. | $[\neg Q, \neg R]$ | β | [4.] |
| 6. | $[\neg Q]$ | α_1 | [2.] |
| 7. | $[\neg(P \supset Q)]$ | α_2 | [2.] |
| 8. | $[\neg\neg P]$ | α_1 | [7.] |
| 9. | $[Q]$ | α_2 | [7.] |
| 10. | $[P]$ | $\neg\neg$ | [9.] |

Remember: Resol. operates on conjunctive sets of disjunctions.

Remember: Propositional Resolution Expansion Rules

$$\frac{\neg\neg Z}{Z} \quad \frac{\neg T}{\perp} \quad \frac{\neg\perp}{T} \quad \frac{\beta}{\beta_1 \quad \beta_2} \quad \frac{\alpha}{\alpha_1 \mid \alpha_2}$$

Example — $\{P \downarrow (Q \wedge R), \neg(Q \vee (P \supset Q))\}$

165

- | | | | |
|-----|--------------------------------|------------|------|
| 1. | $[P \downarrow (Q \wedge R)]$ | | |
| 2. | $[\neg(Q \vee (P \supset Q))]$ | | |
| 3. | $[\neg P]$ | α_1 | [1.] |
| 4. | $[\neg(Q \wedge R)]$ | α_2 | [1.] |
| 5. | $[\neg Q, \neg R]$ | β | [4.] |
| 6. | $[\neg Q]$ | α_1 | [2.] |
| 7. | $[\neg(P \supset Q)]$ | α_2 | [2.] |
| 8. | $[\neg\neg P]$ | α_1 | [7.] |
| 9. | $[Q]$ | α_2 | [7.] |
| 10. | $[P]$ | $\neg\neg$ | [9.] |

Remember: Resol. operates on conjunctive sets of disjunctions.

▶ Just as before:

- ▶ composed of the rules we already know from propositional logic
- ▶ also the resolution rule stays the same
- ▶ but we have two additional rules: γ -rule and δ -rule
- ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
- ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$\frac{\gamma}{\gamma(t)}$ for any closed term t of L^{par}

$\frac{\delta}{\delta(p)}$ for a new parameter p

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$\frac{\gamma}{\gamma(t)}$ for any closed term t of L^{par}

$\frac{\delta}{\delta(p)}$ for a new parameter p

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{par}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{par}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

- ▶ Just as before:
 - ▶ composed of the rules we already know from propositional logic
 - ▶ also the resolution rule stays the same
 - ▶ but we have two additional rules: γ -rule and δ -rule
 - ▶ γ -rule is about universal statements: any closed term from L^{par} can be used
 - ▶ δ -rule is about existential statements: new witnesses are needed and for this we will use symbols that have not been previously introduced (in the resolution construction) — parameters

Definition — First-Order Resolution Expansion Rules

166

$$\frac{\gamma}{\gamma(t)} \text{ for any closed term } t \text{ of } L^{\text{par}}$$

$$\frac{\delta}{\delta(p)} \text{ for a new parameter } p$$

plus all Propositional Resolution Expansion Rules

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ $resolution[7., 9.]$
11. \emptyset $resolution[6., 10.]$

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. \emptyset *resolution*[6., 10.]

Example

167

In order to prove $(\forall x)(P(x) \vee Q(x)) \supset ((\exists x)P(x) \vee (\forall x)Q(x))$ we derive the empty clause from

$$\{\neg[(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]]\}$$

1. $[\neg\{(\forall x)[P(x) \vee Q(x)] \supset [(\exists x)P(x) \vee (\forall x)Q(x)]\}]$
2. $[(\forall x)(P(x) \vee Q(x))]$ $\alpha_1 [1.]$
3. $[\neg((\exists x)P(x) \vee (\forall x)Q(x))]$ $\alpha_2 [1.]$
4. $[\neg(\exists x)P(x)]$ $\alpha_1 [3.]$
5. $[\neg(\forall x)Q(x)]$ $\alpha_2 [3.]$
6. $[\neg Q(c)]$ $\delta(c) [5.]$
7. $[\neg P(c)]$ $\gamma(c) [4.]$
8. $[P(c) \vee Q(c)]$ $\gamma(c) [2.]$
9. $[P(c), Q(c)]$ $\beta[8.]$
10. $[Q(c)]$ *resolution*[7., 9.]
11. $[]$ *resolution*[6., 10.]

Exercises:

- ▶ Redo earlier tableaux example problems with resolution
- ▶ Prove, by structural induction, that an application of the Resolution Rule involving a formula that is not atomic can be replaced by resolutions involving only atomic formulas.

Lemma

168

1. If any Tableau Expansion Rule is applied to a satisfiable tableau, the result is another satisfiable tableau.
2. If any Resolution Expansion Rule, or the Resolution Rule is applied to a satisfiable resolution expansion the result is again satisfiable.

Proof: (1.) The argumentation for the propositional rules is as in the propositional case. For the quantifier rules we apply an earlier Proposition (see 133): Let S be a set of sentences (closed formulas), and γ and δ be sentences.

1. If $S \cup \{\gamma\}$ is satisfiable, so is $S \cup \{\gamma, \gamma(t)\}$ for any closed term t .
 2. If $S \cup \{\delta\}$ is satisfiable, so is $S \cup \{\delta, \delta(p)\}$ for any constant symbol p that is new to S and δ .
- (2.) is analogous, ... exercise ...

Lemma

168

1. If any Tableau Expansion Rule is applied to a satisfiable tableau, the result is another satisfiable tableau.
2. If any Resolution Expansion Rule, or the Resolution Rule is applied to a satisfiable resolution expansion the result is again satisfiable.

Proof: (1.) The argumentation for the propositional rules is as in the propositional case. For the quantifier rules we apply an earlier Proposition (see 133): Let S be a set of sentences (closed formulas), and γ and δ be sentences.

1. If $S \cup \{\gamma\}$ is satisfiable, so is $S \cup \{\gamma, \gamma(t)\}$ for any closed term t .
 2. If $S \cup \{\delta\}$ is satisfiable, so is $S \cup \{\delta, \delta(p)\}$ for any constant symbol p that is new to S and δ .
- (2.) is analogous, ... exercise ...

Lemma

168

1. If any Tableau Expansion Rule is applied to a satisfiable tableau, the result is another satisfiable tableau.
2. If any Resolution Expansion Rule, or the Resolution Rule is applied to a satisfiable resolution expansion the result is again satisfiable.

Proof: (1.) The argumentation for the propositional rules is as in the propositional case. For the quantifier rules we apply an earlier Proposition (see 133): Let S be a set of sentences (closed formulas), and γ and δ be sentences.

1. If $S \cup \{\gamma\}$ is satisfiable, so is $S \cup \{\gamma, \gamma(t)\}$ for any closed term t .
 2. If $S \cup \{\delta\}$ is satisfiable, so is $S \cup \{\delta, \delta(p)\}$ for any constant symbol p that is new to S and δ .
- (2.) is analogous, ... exercise ...

Lemma

168

1. If any Tableau Expansion Rule is applied to a satisfiable tableau, the result is another satisfiable tableau.
2. If any Resolution Expansion Rule, or the Resolution Rule is applied to a satisfiable resolution expansion the result is again satisfiable.

Proof: (1.) The argumentation for the propositional rules is as in the propositional case. For the quantifier rules we apply an earlier Proposition (see 133): Let S be a set of sentences (closed formulas), and γ and δ be sentences.

1. If $S \cup \{\gamma\}$ is satisfiable, so is $S \cup \{\gamma, \gamma(t)\}$ for any closed term t .
2. If $S \cup \{\delta\}$ is satisfiable, so is $S \cup \{\delta, \delta(p)\}$ for any constant symbol p that is new to S and δ .

(2.) is analogous, ... exercise ...

Lemma

168

1. If any Tableau Expansion Rule is applied to a satisfiable tableau, the result is another satisfiable tableau.
2. If any Resolution Expansion Rule, or the Resolution Rule is applied to a satisfiable resolution expansion the result is again satisfiable.

Proof: (1.) The argumentation for the propositional rules is as in the propositional case. For the quantifier rules we apply an earlier Proposition (see 133): Let S be a set of sentences (closed formulas), and γ and δ be sentences.

1. If $S \cup \{\gamma\}$ is satisfiable, so is $S \cup \{\gamma, \gamma(t)\}$ for any closed term t .
 2. If $S \cup \{\delta\}$ is satisfiable, so is $S \cup \{\delta, \delta(p)\}$ for any constant symbol p that is new to S and δ .
- (2.) is analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.

(2.) Analogous, ... exercise ...

Theorem — Soundness

169

1. If X has a tableau proof, then X is valid.
2. If X has a resolution proof, then X is valid.

Proof: (1.) Suppose X has a tableau proof but is not valid. Show contradiction. Since X is not valid, there is a model in which $\neg X$ is true. The construction of the tableau proof begins with the single node $\neg X$; this tableau is obviously satisfiable. By Lemma 168 every subsequent tableau is satisfiable, including the final closed tableau. However, a closed tableau cannot be satisfiable. Contradiction.
(2.) Analogous, . . . exercise . . .

Exercises:

1. Complete the missing parts in Lemma 168 and the Soundness Theorem.

2. Prove strong soundness of the tableau system:

Let X be a sentence and S be a set of sentences of the language \mathbf{L} . Then $S \vdash_{ft} X$ implies $S \models_f X$.

3. Prove strong soundness of the resolution system:

Let X be a sentence and S be a set of sentences of the language \mathbf{L} . Then $S \vdash_{fr} X$ implies $S \models_f X$.

- ▶ We again apply the abstract consistency proof technique.

Definition — Tableau Consistent

170

A finite set S of sentences of \mathbf{L}^{par} is *tableau consistent* if there is no closed tableau for S .

Lemma

171

The collection of all tableau consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof:

The cases ∇_c , ∇_{\perp} , $\nabla_{\neg\neg}$, ∇_{α} are analogous to the proof we have already seen for the propositional case. Also ∇_{γ} , ∇_{δ} are easy. The only slightly tricky case is ∇_{β} :

∇_{β} : if $\beta \in S$ (for $S \in \mathcal{C}$) then $S \cup \{\beta_1\} \in \mathcal{C}$ or $S \cup \{\beta_2\} \in \mathcal{C}$

- ▶ We again apply the abstract consistency proof technique.

Definition — Tableau Consistent

170

A finite set S of sentences of \mathbf{L}^{par} is *tableau consistent* if there is no closed tableau for S .

Lemma

171

The collection of all tableau consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof:

The cases ∇_c , ∇_{\perp} , $\nabla_{\neg\neg}$, ∇_{α} are analogous to the proof we have already seen for the propositional case. Also ∇_{γ} , ∇_{δ} are easy. The only slightly tricky case is ∇_{β} :

∇_{β} : if $\beta \in S$ (for $S \in \mathcal{C}$) then $S \cup \{\beta_1\} \in \mathcal{C}$ or $S \cup \{\beta_2\} \in \mathcal{C}$

- ▶ We again apply the abstract consistency proof technique.

Definition — Tableau Consistent

170

A finite set S of sentences of \mathbf{L}^{par} is *tableau consistent* if there is no closed tableau for S .

Lemma

171

The collection of all tableau consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof:

The cases ∇_c , ∇_{\perp} , $\nabla_{\neg\neg}$, ∇_{α} are analogous to the proof we have already seen for the propositional case. Also ∇_{γ} , ∇_{δ} are easy. The only slightly tricky case is ∇_{β} :

∇_{β} : if $\beta \in S$ (for $S \in \mathcal{C}$) then $S \cup \{\beta_1\} \in \mathcal{C}$ or $S \cup \{\beta_2\} \in \mathcal{C}$

- ▶ We again apply the abstract consistency proof technique.

Definition — Tableau Consistent

170

A finite set S of sentences of \mathbf{L}^{par} is *tableau consistent* if there is no closed tableau for S .

Lemma

171

The collection of all tableau consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof:

The cases ∇_c , ∇_{\perp} , $\nabla_{\neg\neg}$, ∇_{α} are analogous to the proof we have already seen for the propositional case. Also ∇_{γ} , ∇_{δ} are easy. The only slightly tricky case is ∇_{β} :

∇_{β} : if $\beta \in S$ (for $S \in \mathcal{C}$) then $S \cup \{\beta_1\} \in \mathcal{C}$ or $S \cup \{\beta_2\} \in \mathcal{C}$

Proof (cont'd):

∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):

 β X_1 \vdots X_n β_1 $|$ T_1 β_2 $|$ T_2 $\beta [1.]$

Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

**Proof (cont'd):**

∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):

$$\begin{array}{c} \beta \\ X_1 \\ \vdots \\ X_n \end{array}$$

$$\begin{array}{ccc} \beta_1 & \beta_2 & \beta [1.] \\ | & | & | \\ T_1 & T_2 & \end{array}$$

Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

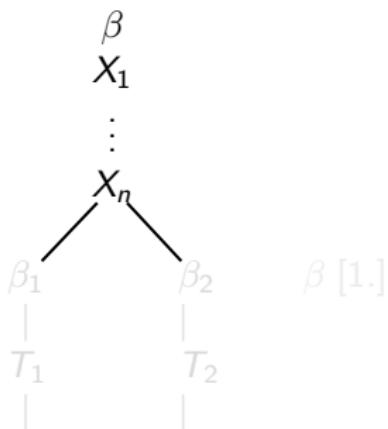
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):

 β X_1 \vdots X_n β_1 $|$ T_1 β_2 $|$ T_2 $\beta [1.]$

Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

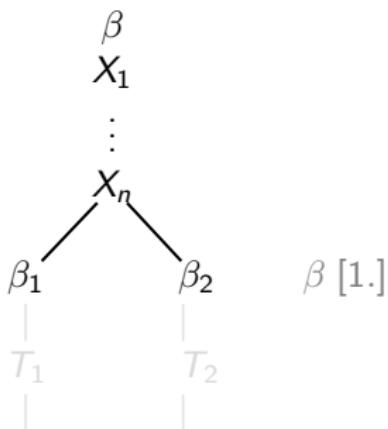
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

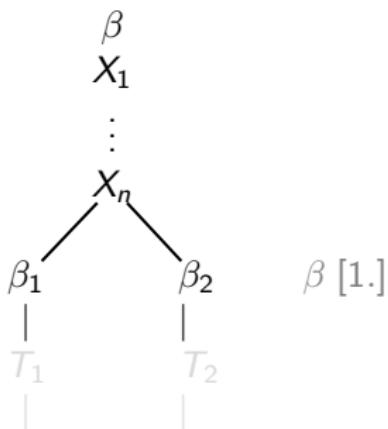
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

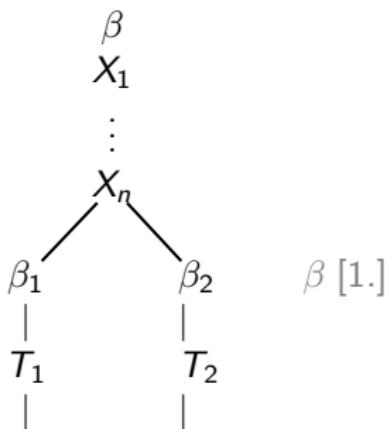
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

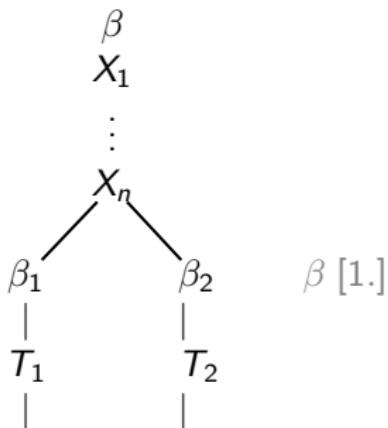
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

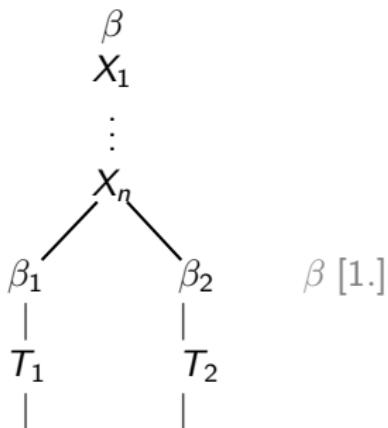
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \text{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

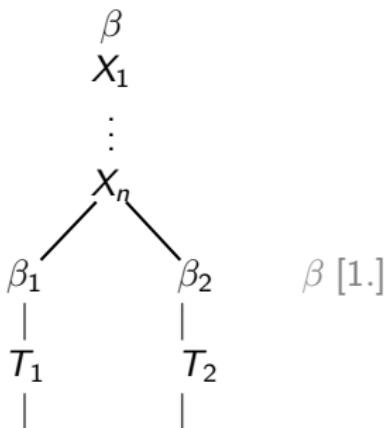
∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \mathbf{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Proof (cont'd):

∇_β : Prove by contraposition — Suppose $\beta \in S$ but neither $S \cup \{\beta_1\}$ nor $S \cup \{\beta_2\}$ is tableau consistent; we show that S is not tableau consistent. From the assumption we know that there is a closed tableau T_1 for $S \cup \{\beta_1\}$ and a closed tableau T_2 for $S \cup \{\beta_2\}$. The idea now is to combine T_1 and T_2 to obtain a closed tableau T for S as follows (assume $S = \{\beta, X_1, \dots, X_n\}$):



Problem: Parameters $c \in \mathbf{par}$ may occur in both T_1 and T_2 — thus, T would not respect the conditions of the δ -rule. Solution: Replace c in T_1 by a parameter d that is new to both T_1 and T_2 . Do this repeatedly for all joint parameters. Easy to verify that T is a proper closed tableau for S . Hence, S is not tableau consistent.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Theorem — Completeness of First-Order Tableaux

172

If the sentence X of \mathbf{L} is valid, then X has a tableau proof.

Proof: By contraposition.

If X does not have a tableau proof, there is no closed tableau for $\{\neg X\}$. Hence, $\{\neg X\}$ is tableau consistent and hence satisfiable by the First-Order Model Existence Theorem. Thus, X is not valid.

Remark: The completeness proof also works if we require *atomically* closed tableau.

Remarks:

- ▶ proof idea as expected (by now): use abstract consistency
- ▶ the propositional cases are similar/identical to propositional resolution
- ▶ other cases are similar to first-order tableau
- ▶ only problematic case is again ∇_β where a similar renaming as sketched above is required

Remember: X -enlargement

Let $X, A_1, \dots, A_n \in \mathbf{P}$. $[X, A_1, \dots, A_n]$ and $[A_1, \dots, A_n]$ are both called *X -enlargements* of $[A_1, \dots, A_n]$.

If S is a set of disjunctions and S^* is the result of replacing each member of S by an X -enlargement, then S^* is called an *X -enlargement* of S .

Remarks:

- ▶ proof idea as expected (by now): use abstract consistency
- ▶ the propositional cases are similar/identical to propositional resolution
- ▶ other cases are similar to first-order tableau
- ▶ only problematic case is again ∇_β where a similar renaming as sketched above is required

Remember: *X*-enlargement

Let $X, A_1, \dots, A_n \in \mathbf{P}$. $[X, A_1, \dots, A_n]$ and $[A_1, \dots, A_n]$ are both called *X-enlargements* of $[A_1, \dots, A_n]$.

If S is a set of disjunctions and S^* is the result of replacing each member of S by an *X*-enlargement, then S^* is called an *X-enlargement* of S .

Remarks:

- ▶ proof idea as expected (by now): use abstract consistency
- ▶ the propositional cases are similar/identical to propositional resolution
- ▶ other cases are similar to first-order tableau
- ▶ only problematic case is again ∇_β where a similar renaming as sketched above is required

Remember: *X*-enlargement

Let $X, A_1, \dots, A_n \in \mathbf{P}$. $[X, A_1, \dots, A_n]$ and $[A_1, \dots, A_n]$ are both called *X-enlargements* of $[A_1, \dots, A_n]$.

If S is a set of disjunctions and S^* is the result of replacing each member of S by an *X*-enlargement, then S^* is called an *X-enlargement* of S .

Lemma

173

Suppose S_1 and S_2 are sets of disjunctions, and S_2 is an X -enlargement of S_1 . If the disjunction $D_1 = [A_1, \dots, A_n]$ is resolution derivable from S_1 , then there is an X -enlargement D_2 of D_1 that is resolution derivable from S_2 , provided X contains no parameters occurring in the derivation of D_1 .

Proof: ... like in propositional case, exercise ...

Definition — Resolution Consistent

174

A finite set S of sentences of \mathbf{L}^{par} is *resolution consistent* if there is no closed resolution expansion for S .

Lemma

173

Suppose S_1 and S_2 are sets of disjunctions, and S_2 is an X -enlargement of S_1 . If the disjunction $D_1 = [A_1, \dots, A_n]$ is resolution derivable from S_1 , then there is an X -enlargement D_2 of D_1 that is resolution derivable from S_2 , provided X contains no parameters occurring in the derivation of D_1 .

Proof: . . . like in propositional case, exercise . . .

Definition — Resolution Consistent

174

A finite set S of sentences of \mathbf{L}^{par} is *resolution consistent* if there is no closed resolution expansion for S .

Lemma

173

Suppose S_1 and S_2 are sets of disjunctions, and S_2 is an X -enlargement of S_1 . If the disjunction $D_1 = [A_1, \dots, A_n]$ is resolution derivable from S_1 , then there is an X -enlargement D_2 of D_1 that is resolution derivable from S_2 , provided X contains no parameters occurring in the derivation of D_1 .

Proof: . . . like in propositional case, exercise . . .

Definition — Resolution Consistent

174

A finite set S of sentences of \mathbf{L}^{par} is *resolution consistent* if there is no closed resolution expansion for S .

Lemma

175

The collection of all resolution consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof: ... like in propositional case, except that we have to apply a renaming similar to above ... exercise ...

Theorem — Completeness for First-Order Resolution

176

If the sentence X of \mathbf{L} is valid, then X has a resolution proof.

Proof: ... analogous to the tableaux case, exercise ...

Lemma

175

The collection of all resolution consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof: ... like in propositional case, except that we have to apply a renaming similar to above ... exercise ...

Theorem — Completeness for First-Order Resolution

176

If the sentence X of \mathbf{L} is valid, then X has a resolution proof.

Proof: ... analogous to the tableaux case, exercise ...

Lemma

175

The collection of all resolution consistent sets of sentences of \mathbf{L}^{par} is a first-order consistency property.

Proof: ... like in propositional case, except that we have to apply a renaming similar to above ... exercise ...

Theorem — Completeness for First-Order Resolution

176

If the sentence X of \mathbf{L} is valid, then X has a resolution proof.

Proof: ... analogous to the tableaux case, exercise ...

Exercises:

1. Complete the proofs above.

2. Prove strong completeness of the tableau system:

Let X be a sentence and S be a set of sentences of the language \mathbf{L} . Then $S \models_f X$ implies $S \vdash_{ft} X$.

3. Prove strong completeness of the resolution system:

Let X be a sentence and S be a set of sentences of the language \mathbf{L} . Then $S \models_f X$ implies $S \vdash_{fr} X$.

The above calculi — semantic tableaux and resolution — lead to very inefficient implementations:

- ▶ Problem: the γ -rule allows us to introduce *any closed term!* So which one(s) do we choose? Enumerating them all is a bad option for practice!
- ▶ Solution: postpone the search and work with *free variables instead*; then use *unification* during proof search to determine the 'right' closed terms to use.

But then there arises another issue:

- ▶ Problem: the δ -rule now does not know *which parameters are new*, since we have delayed the introduction new closed terms in the γ -rule.
- ▶ Solution: use *Skolem terms* in the δ -rule — from δ derive $\delta(f(x_1, \dots, x_n))$ where f is new function parameter and were x_1, \dots, x_n are the free variables introduced so far. Note that this 'forces' the term to be 'new'.

Thus: Tableaux and resolution from now on support free variables.

The above calculi — semantic tableaux and resolution — lead to very inefficient implementations:

- ▶ Problem: the γ -rule allows us to introduce *any closed term!* So which one(s) do we choose? Enumerating them all is a bad option for practice!
- ▶ Solution: postpone the search and work with *free variables instead*; then use *unification* during proof search to determine the 'right' closed terms to use.

But then there arises another issue:

- ▶ Problem: the δ -rule now does not know *which parameters are new*, since we have delayed the introduction new closed terms in the γ -rule.
- ▶ Solution: use *Skolem terms* in the δ -rule — from δ derive $\delta(f(x_1, \dots, x_n))$ where f is new function parameter and were x_1, \dots, x_n are the free variables introduced so far. Note that this 'forces' the term to be 'new'.

Thus: Tableaux and resolution from now on support free variables.

The above calculi — semantic tableaux and resolution — lead to very inefficient implementations:

- ▶ Problem: the γ -rule allows us to introduce *any closed term!* So which one(s) do we choose? Enumerating them all is a bad option for practice!
- ▶ Solution: postpone the search and work with *free variables instead*; then use *unification* during proof search to determine the 'right' closed terms to use.

But then there arises another issue:

- ▶ Problem: the δ -rule now does not know *which parameters are new*, since we have delayed the introduction new closed terms in the γ -rule.
- ▶ Solution: use *Skolem terms* in the δ -rule — from δ derive $\delta(f(x_1, \dots, x_n))$ where f is new function parameter and were x_1, \dots, x_n are the free variables introduced so far. Note that this 'forces' the term to be 'new'.

Thus: Tableaux and resolution from now on support free variables.

The above calculi — semantic tableaux and resolution — lead to very inefficient implementations:

- ▶ Problem: the γ -rule allows us to introduce *any closed term!* So which one(s) do we choose? Enumerating them all is a bad option for practice!
- ▶ Solution: postpone the search and work with *free variables instead*; then use *unification* during proof search to determine the 'right' closed terms to use.

But then there arises another issue:

- ▶ Problem: the δ -rule now does not know *which parameters are new*, since we have delayed the introduction new closed terms in the γ -rule.
- ▶ Solution: use *Skolem terms* in the δ -rule — from δ derive $\delta(f(x_1, \dots, x_n))$ where f is new function parameter and were x_1, \dots, x_n are the free variables introduced so far. Note that this 'forces' the term to be 'new'.

Thus: Tableaux and resolution from now on support free variables.

The above calculi — semantic tableaux and resolution — lead to very inefficient implementations:

- ▶ Problem: the γ -rule allows us to introduce *any closed term!* So which one(s) do we choose? Enumerating them all is a bad option for practice!
- ▶ Solution: postpone the search and work with *free variables instead*; then use *unification* during proof search to determine the 'right' closed terms to use.

But then there arises another issue:

- ▶ Problem: the δ -rule now does not know *which parameters are new*, since we have delayed the introduction new closed terms in the γ -rule.
- ▶ Solution: use *Skolem terms* in the δ -rule — from δ derive $\delta(f(x_1, \dots, x_n))$ where f is new function parameter and were x_1, \dots, x_n are the free variables introduced so far. Note that this 'forces' the term to be 'new'.

Thus: Tableaux and resolution from now on support free variables.

Example — $(\forall x)(\forall y)R(g(x, a), g(b, y)) \supset (\exists u)(\exists v)R(u, v)$ **177**

... exercise ... discuss example on blackboard ...

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Anteijer, Jacques Herbrand: Life, Logic, and Automated Deduction, Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann, Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Anteijer, Jacques Herbrand: Life, Logic, and Automated Deduction, Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann, Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier, Jacques Herbrand: Life, Logic, and Automated Deduction. Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann. Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier, Jacques Herbrand: Life, Logic, and Automated Deduction. Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann. Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier, Jacques Herbrand: Life, Logic, and Automated Deduction. Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann. Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier, Jacques Herbrand: Life, Logic, and Automated Deduction. Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann. Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

- ▶ Given two terms t_1 and t_2 of language \mathbf{L}^{par} ; both terms may have free variable occurrences, let's say the free variables of t_1 are u_1, \dots, u_n and the free variables of t_2 are v_1, \dots, v_m .
- ▶ Can we instantiate u_1, \dots, u_n and v_1, \dots, v_m with terms from \mathbf{L}^{par} in such a way that t_1 and t_2 become (syntactically) equal.
- ▶ Papers on unification:
 - ▶ Jacques Herbrand, Investigations in proof theory, 1930. (For an overview on Herbrand's work see: C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier, Jacques Herbrand: Life, Logic, and Automated Deduction. Handbook of the History of Logic, Volume 5, 2009.)
 - ▶ J. A. Robinson, A machine-oriented logic based on the resolution principle, Journal of the ACM 12, 1965.
 - ▶ F.Baader and J.Siekmann. Unification theory, Handbook of Logic in Artificial Intelligence and Logic Programming, 1994.
 - ▶ F.Baader and W.Snyder, Unification theory, Handbook of Automated Reasoning, 2001.

Definition — More General Substitution

178

Let σ_1 and σ_2 be substitutions. We say σ_2 is more general than σ_1 if, for some substitution τ , $\sigma_1 = \sigma_2\tau$.

Example

179

1. Show that $\sigma_2 = \{x/f(g(x, y)), y/g(z, b)\}$ is more general than $\sigma_1 = \{x/f(g(a, h(z))), y/g(h(x), b), x/h(x)\}$.
2. Is σ_1 more general than σ_1 ?

Definition — More General Substitution

178

Let σ_1 and σ_2 be substitutions. We say σ_2 is more general than σ_1 if, for some substitution τ , $\sigma_1 = \sigma_2\tau$.

Example

179

1. Show that $\sigma_2 = \{x/f(g(x, y)), y/g(z, b)\}$ is more general than $\sigma_1 = \{x/f(g(a, h(z))), y/g(h(x), b), x/h(x)\}$.
2. Is σ_1 more general than σ_1 ?

Proposition — Transitivity of 'More general'

180

If σ_3 is more general than σ_2 and σ_2 is more general than σ_1 , then σ_3 is more general than σ_1 .

Proof: We know $\sigma_1 = \sigma_2\tau$ and $\sigma_2 = \sigma_3\theta$.

But then $\sigma_1 = \sigma_2\tau = (\sigma_3\theta)\tau = \sigma_3(\theta\tau)$.

Definition — Unifier/Most General Unifier (MGU)

181

Let t_1 and t_2 be terms. A substitution σ is a *unifier for t_1 and t_2* if $t_1\sigma = t_2\sigma$. t_1 and t_2 are *unifiable* if they have a unifier. A substitution is a *most general unifier MGU (of t_1 and t_2)* if it is a unifier and more general than any other unifier of t_1 and t_2 .
(These notions do extend to sets of terms in the obvious way).

Proposition — Transitivity of 'More general'

180

If σ_3 is more general than σ_2 and σ_2 is more general than σ_1 , then σ_3 is more general than σ_1 .

Proof: We know $\sigma_1 = \sigma_2\tau$ and $\sigma_2 = \sigma_3\theta$.

But then $\sigma_1 = \sigma_2\tau = (\sigma_3\theta)\tau = \sigma_3(\theta\tau)$.

Definition — Unifier/Most General Unifier (MGU)

181

Let t_1 and t_2 be terms. A substitution σ is a *unifier for t_1 and t_2* if $t_1\sigma = t_2\sigma$. t_1 and t_2 are *unifiable* if they have a unifier. A substitution is a *most general unifier MGU (of t_1 and t_2)* if it is a unifier and more general than any other unifier of t_1 and t_2 .
(These notions do extend to sets of terms in the obvious way).

Proposition — Transitivity of 'More general'

180

If σ_3 is more general than σ_2 and σ_2 is more general than σ_1 , then σ_3 is more general than σ_1 .

Proof: We know $\sigma_1 = \sigma_2\tau$ and $\sigma_2 = \sigma_3\theta$.

But then $\sigma_1 = \sigma_2\tau = (\sigma_3\theta)\tau = \sigma_3(\theta\tau)$.

Definition — Unifier/Most General Unifier (MGU)

181

Let t_1 and t_2 be terms. A substitution σ is a *unifier for t_1 and t_2* if $t_1\sigma = t_2\sigma$. t_1 and t_2 are *unifiable* if they have a unifier. A substitution is a *most general unifier MGU (of t_1 and t_2)* if it is a unifier and more general than any other unifier of t_1 and t_2 .
(These notions do extend to sets of terms in the obvious way).

Proposition — Transitivity of 'More general'

180

If σ_3 is more general than σ_2 and σ_2 is more general than σ_1 , then σ_3 is more general than σ_1 .

Proof: We know $\sigma_1 = \sigma_2\tau$ and $\sigma_2 = \sigma_3\theta$.

But then $\sigma_1 = \sigma_2\tau = (\sigma_3\theta)\tau = \sigma_3(\theta\tau)$.

Definition — Unifier/Most General Unifier (MGU)

181

Let t_1 and t_2 be terms. A substitution σ is a *unifier for t_1 and t_2* if $t_1\sigma = t_2\sigma$. t_1 and t_2 are *unifiable* if they have a unifier. A substitution is a *most general unifier MGU (of t_1 and t_2)* if it is a unifier and more general than any other unifier of t_1 and t_2 .
(These notions do extend to sets of terms in the obvious way).

Example

182

$f(y, h(a))$ and $f(h(x), h(z))$ unifiable with

1. $\{y/h(x), z/a\}$.
2. $\{x/k(w), y/h(k(w)), z/a\}$.

Which one is more general?

Note: Technically, two terms t_1 and t_2 may have more than just one most general unifier (consider $g(x, x)$ and $g(y, z)$), but if so then they are the same up to a variable renaming.

Example

182

$f(y, h(a))$ and $f(h(x), h(z))$ unifiable with

1. $\{y/h(x), z/a\}$.
2. $\{x/k(w), y/h(k(w)), z/a\}$.

Which one is more general?

Note: Technically, two terms t_1 and t_2 may have more than just one most general unifier (consider $g(x, x)$ and $g(y, z)$), but if so then they are the same up to a variable renaming.

Definition — Variable Renaming

183

A substitution η is a *variable renaming* for a set V of variables if

1. For each $x \in V$, $x\eta$ is a variable.
2. For $x, y \in V$ with $x \neq y$, $x\eta$ and $y\eta$ are distinct.

Definition — Variable Range

184

The *variable range* for a substitution σ is the set of variables that occur in terms of the forms $x\sigma$, where x is a variable.

Proposition — Most General Unifiers

185

Suppose both σ_1 and σ_2 are most general unifiers of t_1 and t_2 . Then there is a variable renaming η for the variable range of σ such that $\sigma_1\eta = \sigma_2$.

Proof: ... straightforward, not here ...

Definition — Variable Renaming

183

A substitution η is a *variable renaming* for a set V of variables if

1. For each $x \in V$, $x\eta$ is a variable.
2. For $x, y \in V$ with $x \neq y$, $x\eta$ and $y\eta$ are distinct.

Definition — Variable Range

184

The *variable range* for a substitution σ is the set of variables that occur in terms of the forms $x\sigma$, where x is a variable.

Proposition — Most General Unifiers

185

Suppose both σ_1 and σ_2 are most general unifiers of t_1 and t_2 . Then there is a variable renaming η for the variable range of σ such that $\sigma_1\eta = \sigma_2$.

Proof: ... straightforward, not here ...

Definition — Variable Renaming

183

A substitution η is a *variable renaming* for a set V of variables if

1. For each $x \in V$, $x\eta$ is a variable.
2. For $x, y \in V$ with $x \neq y$, $x\eta$ and $y\eta$ are distinct.

Definition — Variable Range

184

The *variable range* for a substitution σ is the set of variables that occur in terms of the forms $x\sigma$, where x is a variable.

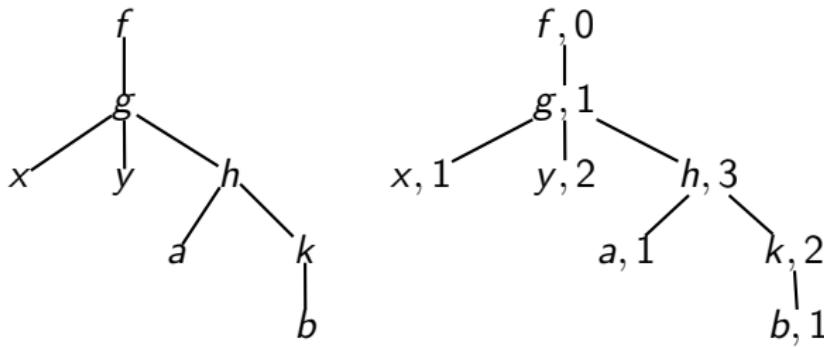
Proposition — Most General Unifiers

185

Suppose both σ_1 and σ_2 are most general unifiers of t_1 and t_2 .
Then there is a variable renaming η for the variable range of σ
such that $\sigma_1\eta = \sigma_2$.

Proof: ... straightforward, not here ...

Augmented Tree Representation for: $f(g(x, y, h(a, k(b))))$.



Allows us to talk about paths through a term, e.g. $\langle f, 0 \rangle, \langle h, 2 \rangle$

Definition — Disagreement Pair

186

A *disagreement pair* for terms t_1 and t_2 is a pair of terms $[d_1, d_2]$, such that

- ▶ d_1 is a subterm of t_1 and d_2 is a subterm of t_2 , and
- ▶ thinking of terms as augmented trees, d_1 and d_2 have distinct labels at their roots,
- ▶ while the path from the root of t_1 down to the root of d_1 and the path from the root of t_2 down to the root of d_2 are the same.

Definition — Disagreement Pair

186

A *disagreement pair* for terms t_1 and t_2 is a pair of terms $[d_1, d_2]$, such that

- ▶ d_1 is a subterm of t_1 and d_2 is a subterm of t_2 , and
- ▶ thinking of terms as augmented trees, d_1 and d_2 have distinct labels at their roots,
- ▶ while the path from the root of t_1 down to the root of d_1 and the path from the root of t_2 down to the root of d_2 are the same.

Definition — Disagreement Pair

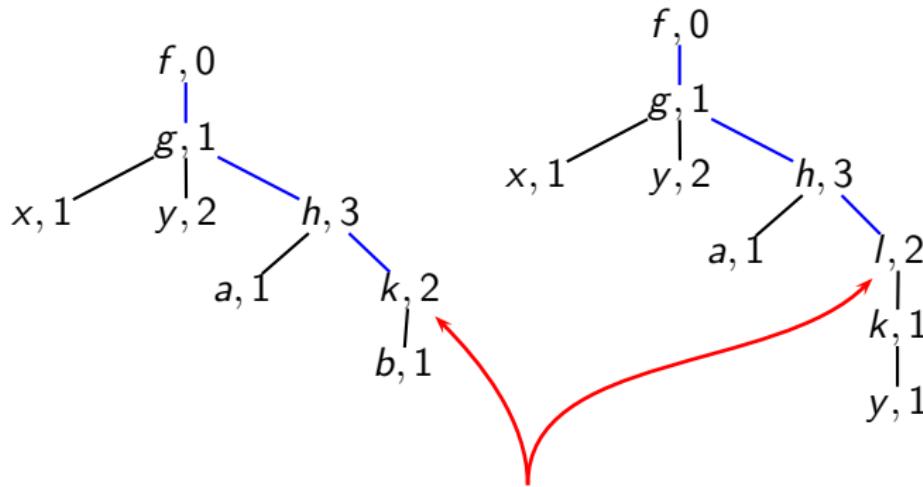
186

A *disagreement pair* for terms t_1 and t_2 is a pair of terms $[d_1, d_2]$, such that

- ▶ d_1 is a subterm of t_1 and d_2 is a subterm of t_2 , and
- ▶ thinking of terms as augmented trees, d_1 and d_2 have distinct labels at their roots,
- ▶ while the path from the root of t_1 down to the root of d_1 and the path from the root of t_2 down to the root of d_2 are the same.

Disagreement Pair for terms

$f(g(x, y, h(a, k(b))))$ and $f(g(x, y, h(a, l(k(y))))).$



Disagreement pair:

$[k(b), l(k(y))]$

Unification Algorithm (Robinson)

```
Let  $\sigma := \epsilon$ ;  
While  $t_1\sigma \neq t_2\sigma$  do  
begin  
choose a disagreement pair  $[d_1, d_2]$  for  $t_1\sigma$  and  $t_2\sigma$ ;  
if neither  $d_1$  nor  $d_2$  is a variable then FAIL;  
let  $x$  be whichever of  $d_1$  and  $d_2$  is a variable  
(if both are, choose one)  
and let  $t$  be the other one of  $d_1, d_2$ ;  
if  $x$  occurs in  $t$  then FAIL;  
let  $\sigma := \sigma\{x/t\}$ ;  
end.
```

Theorem — Unification Theorem

187

Given two terms t_1 and t_2 .

- ▶ If t_1 and t_2 are not unifiable, then the Unification Algorithm will FAIL.
- ▶ If t_1 and t_2 are unifiable, then the Unification Algorithm will terminate without FAILure and the final value of σ will be a most general unifier of t_1 and t_2 .

Proof: ... not here ...

Definition — Idempotent Substitution

188

A substitution σ is called *idempotent* if $\sigma = \sigma\sigma$

Corollary

189

If t_1 and t_2 are unifiable, the Unification Algorithm terminates with a final value that is an idempotent most general unifier for them.

Theorem — Unification Theorem

187

Given two terms t_1 and t_2 .

- ▶ If t_1 and t_2 are not unifiable, then the Unification Algorithm will FAIL.
- ▶ If t_1 and t_2 are unifiable, then the Unification Algorithm will terminate without FAILure and the final value of σ will be a most general unifier of t_1 and t_2 .

Proof: ... not here ...

Definition — Idempotent Substitution

188

A substitution σ is called *idempotent* if $\sigma = \sigma\sigma$

Corollary

189

If t_1 and t_2 are unifiable, the Unification Algorithm terminates with a final value that is an idempotent most general unifier for them.

Theorem — Unification Theorem

187

Given two terms t_1 and t_2 .

- ▶ If t_1 and t_2 are not unifiable, then the Unification Algorithm will FAIL.
- ▶ If t_1 and t_2 are unifiable, then the Unification Algorithm will terminate without FAILURE and the final value of σ will be a most general unifier of t_1 and t_2 .

Proof: ... not here ...

Definition — Idempotent Substitution

188

A substitution σ is called *idempotent* if $\sigma = \sigma\sigma$

Corollary

189

If t_1 and t_2 are unifiable, the Unification Algorithm terminates with a final value that is an idempotent most general unifier for them.

Theorem — Unification Theorem

187

Given two terms t_1 and t_2 .

- ▶ If t_1 and t_2 are not unifiable, then the Unification Algorithm will FAIL.
- ▶ If t_1 and t_2 are unifiable, then the Unification Algorithm will terminate without FAILURE and the final value of σ will be a most general unifier of t_1 and t_2 .

Proof: ... not here ...

Definition — Idempotent Substitution

188

A substitution σ is called *idempotent* if $\sigma = \sigma\sigma$

Corollary

189

If t_1 and t_2 are unifiable, the Unification Algorithm terminates with a final value that is an idempotent most general unifier for them.

Idempotent most general unifiers have some nice features, e.g.:

Proposition

190

Suppose σ is an idempotent most general unifier for t_1 and t_2 , and τ is any unifier. Then $\tau = \sigma\tau$.

Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.



Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.



Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.



Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.

Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.

Multiple Unification as a sequence of binary unifications:

Given: set of terms $\{t_0, t_1, t_2, \dots, t_n\}$

Unifier: substitution σ such that $t_0\sigma = t_1\sigma = t_2\sigma = \dots = t_n\sigma$

Most general unifier: one that is more general than any other unifier

Suppose $\{t_0, t_1, t_2, \dots, t_n\}$ has a unifier. Then the computation of a most general unifier for this set of terms can be reduced to a sequence of binary unification problems as follows:

- σ_1 : idempotent most general unifier of t_0 and t_1
- σ_2 : idempotent most general unifier of $t_0\sigma_1$ and $t_2\sigma_1$
- σ_3 : idempotent most general unifier of $t_0\sigma_2$ and $t_3\sigma_2$
- ...
- σ_n : idempotent most general unifier of $t_0\sigma_{n-1}$ and $t_n\sigma_{n-1}$

Then, $\sigma := \sigma_1\sigma_2\sigma_3\dots\sigma_n$ is a MGU of $\{t_0, t_1, t_2, \dots, t_n\}$.

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Given: multi-set $E := \{s_1 = t_1, \dots, s_n = t_n\}$

Unifier of E : substitution σ such that $s_i\sigma = t_i\sigma$ (for all $1 \leq i \leq n$)

Unification after Martinelli/Montanari

$$t = t, E \xrightarrow{\text{mm}} E$$

$$f(s_1, \dots, s_n) = f(t_1, \dots, t_n), E \xrightarrow{\text{mm}} s_1 = t_1, \dots, s_n = t_n, E$$

$$f(\dots) = g(\dots) \xrightarrow{\text{mm}} \text{FAIL}$$

$$x = t, E \xrightarrow{\text{mm}} x = t, E\{x/t\} \quad (\text{if } x \text{ does not occur free in } t)$$

$$x = t, E \xrightarrow{\text{mm}} \text{FAIL} \quad (\text{if } x \text{ occurs free in } t)$$

$$t = x, E \xrightarrow{\text{mm}} x = t, E \quad (\text{if } t \text{ is not a variable})$$

Definition — Solved Form**191**

If $E := \{x_1 = t_1, \dots, x_n = t_n\}$, with x_i being pairwise distinct variables and where x_i does not occur in the free variables of t_i , then E is called in *solved form* representing a solution $\sigma_E = \{x_1/t_1, \dots, x_n/t_n\}$.

Theorem**192**

If E is in solved form then σ_E is a most general unifier of E .

Theorem**193**

1. If $E \rightarrow_{mm} E'$ then σ is a unifier of E iff σ is a unifier of E'
2. If $E \rightarrow_{mm}^* FAIL$ then E is not unifiable.
3. If $E \rightarrow_{mm}^* E'$ with E' in solved form, then σ_E is a MGU of E

Definition — Solved Form**191**

If $E := \{x_1 = t_1, \dots, x_n = t_n\}$, with x_i being pairwise distinct variables and where x_i does not occur in the free variables of t_i , then E is called in *solved form* representing a solution $\sigma_E = \{x_1/t_1, \dots, x_n/t_n\}$.

Theorem**192**

If E is in solved form then σ_E is a most general unifier of E .

Theorem**193**

1. If $E \rightarrow_{mm} E'$ then σ is a unifier of E iff σ is a unifier of E'
2. If $E \rightarrow_{mm}^* FAIL$ then E is not unifiable.
3. If $E \rightarrow_{mm}^* E'$ with E' in solved form, then σ_E is a MGU of E

Definition — Solved Form**191**

If $E := \{x_1 = t_1, \dots, x_n = t_n\}$, with x_i being pairwise distinct variables and where x_i does not occur in the free variables of t_i , then E is called in *solved form* representing a solution $\sigma_E = \{x_1/t_1, \dots, x_n/t_n\}$.

Theorem**192**

If E is in solved form then σ_E is a most general unifier of E .

Theorem**193**

1. If $E \xrightarrow{\text{mm}} E'$ then σ is a unifier of E iff σ is a unifier of E'
2. If $E \xrightarrow{\text{mm}}^* \text{FAIL}$ then E is not unifiable.
3. If $E \xrightarrow{\text{mm}}^* E'$ with E' in solved form, then σ_E is a MGU of E

Some Literature

- ▶ **Paterson, Wegman: Linear Unification, JCSS 17, 1978**
Unifiability is decidable in linear time. A most general unifier can be computed in linear time.
- ▶ **Dwork, Kanellakis, Mitchell: On the sequential nature of unification, J.Log.Progr. 1, 1984**
Unifiability is log-space complete for P, that is, every problem in P can be reduced in log-space to a unifiability problem.
Thus, most likely, unifiability cannot be efficiently parallelized.
- ▶ **Baader, Nipkow: Term rewriting and all that. 1998.**
A very good introduction and overview.

Definition — Skolem Function Symbols

194

Let $\mathbf{L} = \mathbf{L}(\mathbf{R}, \mathbf{F}, \mathbf{C})$ be a first-order language. Let **par** be a countable set of constant symbols not in \mathbf{C} (as before). Let **sko** be a countable set of function symbols not in \mathbf{F} , including infinitely many one-place, infinitely many two-place, and so on. These new function symbols are called *Skolem function symbols*. By \mathbf{L}^{sko} we mean the first-order language $\mathbf{L}(\mathbf{R}, \mathbf{F} \cup \mathbf{sko}, \mathbf{C} \cup \mathbf{par})$.

Remarks:

- ▶ Similar to before, proofs will be about sentences of \mathbf{L} but these proofs will use formulas from \mathbf{L}^{sko} .
- ▶ The tableau system we will introduce now will still leave a great amount of freedom and flexibility on proof creation; for an implementation further restrictions are needed.

Definition — Skolem Function Symbols

194

Let $\mathbf{L} = \mathbf{L}(\mathbf{R}, \mathbf{F}, \mathbf{C})$ be a first-order language. Let **par** be a countable set of constant symbols not in \mathbf{C} (as before). Let **sko** be a countable set of function symbols not in \mathbf{F} , including infinitely many one-place, infinitely many two-place, and so on. These new function symbols are called *Skolem function symbols*. By \mathbf{L}^{sko} we mean the first-order language $\mathbf{L}(\mathbf{R}, \mathbf{F} \cup \text{sko}, \mathbf{C} \cup \text{par})$.

Remarks:

- ▶ Similar to before, proofs will be about sentences of \mathbf{L} but these proofs will use formulas from \mathbf{L}^{sko} .
- ▶ The tableau system we will introduce now will still leave a great amount of freedom and flexibility on proof creation; for an implementation further restrictions are needed.

Definition — Free-Variable Tableau Expansion Rules 195

$$\frac{\gamma}{\gamma(x)} \text{ (}x \text{ new variable)}$$
$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (}f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta\text{)}$$

Definition — Substitution is Free for T 196

Let σ be a substitution and T be a tableau. We extend σ to T by setting $T\sigma$ to be the result of replacing each formula X in T by $X\sigma$. We say σ is *free for T* , provided that σ is free for every formula in T .

Definition — Tableau Substitution Rule 197

If T is a tableau for a set S of sentences of L and if the substitution σ is free for T , then $T\sigma$ is also a tableau for S .

Definition — Free-Variable Tableau Expansion Rules

195

$$\frac{\gamma}{\gamma(x)} \text{ (} x \text{ new variable)}$$

$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (} f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta)$$

Definition — Substitution is Free for T

196

Let σ be a substitution and T be a tableau. We extend σ to T by setting $T\sigma$ to be the result of replacing each formula X in T by $X\sigma$. We say σ is *free for T* , provided that σ is free for every formula in T .

Definition — Tableau Substitution Rule

197

If T is a tableau for a set S of sentences of L and if the substitution σ is free for T , then $T\sigma$ is also a tableau for S .

Definition — Free-Variable Tableau Expansion Rules

195

$$\frac{\gamma}{\gamma(x)} \text{ (} x \text{ new variable)}$$

$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (} f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta)$$

Definition — Substitution is Free for T

196

Let σ be a substitution and T be a tableau. We extend σ to T by setting $T\sigma$ to be the result of replacing each formula X in T by $X\sigma$. We say σ is *free for T* , provided that σ is free for every formula in T .

Definition — Tableau Substitution Rule

197

If T is a tableau for a set S of sentences of \mathbf{L} and if the substitution σ is free for T , then $T\sigma$ is also a tableau for S .

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ **198**

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ **198**

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ $\alpha_1 [1.]$
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ $\alpha_2 [1.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ **198**

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ $\alpha_1 [1.]$
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ $\alpha_2 [1.]$
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a) [2.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]
6. $\neg(\exists y)R(b(v_1), v_1, y)$ $\delta(b(v_1))$ [5.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]
6. $\neg(\exists y)R(b(v_1), v_1, y)$ $\delta(b(v_1))$ [5.]
7. $R(v_2, a, f(v_2, a))$ $\gamma(v_2)$ [4.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]
6. $\neg(\exists y)R(b(v_1), v_1, y)$ $\delta(b(v_1))$ [5.]
7. $R(v_2, a, f(v_2, a))$ $\gamma(v_2)$ [4.]
8. $\neg R(b(v_1), v_1, v_3)$ $\gamma(v_3)$ [6.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]
6. $\neg(\exists y)R(b(v_1), v_1, y)$ $\delta(b(v_1))$ [5.]
7. $R(v_2, a, f(v_2, a))$ $\gamma(v_2)$ [4.]
8. $\neg R(b(v_1), v_1, v_3)$ $\gamma(v_3)$ [6.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
5. $\neg(\forall x)(\exists y)R(x, v_1, y)$ $\gamma(v_1)$ [3.]
6. $\neg(\exists y)R(b(v_1), v_1, y)$ $\delta(b(v_1))$ [5.]
7. $R(v_2, a, f(v_2, a))$ $\gamma(v_2)$ [4.]
8. $\neg R(b(v_1), v_1, v_3)$ $\gamma(v_3)$ [6.]

**Now we apply the Tableau Substitution Rule with
 $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ which is free for T .**

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
- 5'. $\neg(\forall x)(\exists y)R(x, a, y)$ $\gamma(a)$ [3.]
- 6'. $\neg(\exists y)R(b(a), a, y)$ $\delta(b(a))$ [5.]
- 7'. $R(b(a), a, f(b(a), a))$ $\gamma(b(a))$ [4.]
- 8'. $\neg R(b(a), a, f(b(a), a))$ $\gamma(f(b(a), a))$ [6.]

Now we apply the Tableau Substitution Rule with
 $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ **which is free for T.**

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
 2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
 3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
 4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
 - 5'. $\neg(\forall x)(\exists y)R(x, a, y)$ $\gamma(a)$ [3.]
 - 6'. $\neg(\exists y)R(b(a), a, y)$ $\delta(b(a))$ [5.]
 - 7'. $R(b(a), a, f(b(a), a))$ $\gamma(b(a))$ [4.]
 - 8'. $\neg R(b(a), a, f(b(a), a))$ $\gamma(f(b(a), a))$ [6.]
-] conflict!

**Now we apply the Tableau Substitution Rule with
 $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ which is free for T .**

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 198

1. $\neg((\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y))$
 2. $(\exists w)(\forall x)R(x, w, f(x, w))$ α_1 [1.]
 3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)$ α_2 [1.]
 4. $(\forall x)R(x, a, f(x, a))$ $\delta(a)$ [2.]
 - 5'. $\neg(\forall x)(\exists y)R(x, a, y)$ $\gamma(a)$ [3.]
 - 6'. $\neg(\exists y)R(b(a), a, y)$ $\delta(b(a))$ [5.]
 - 7'. $R(b(a), a, f(b(a), a))$ $\gamma(b(a))$ [4.]
 - 8'. $\neg R(b(a), a, f(b(a), a))$ $\gamma(f(b(a), a))$ [6.]
-] conflict!

**Now we apply the Tableau Substitution Rule with
 $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ which is free for T .**

Questions:

- How can the substitution σ be determined? — use unification
- When/where to apply unification? — restrict to atomic literals
- This also avoids expensive freeness checks. — Can you see why?

MGU stands for most general unifier.

Definition — MGU Atomic Closure Rule

199

Suppose T is a tableau for a set S of sentences of \mathbf{L} and that some branch of T contains atomic literals A and $\neg B$. Then $T\sigma$ is also a tableau for S , where σ is the most general unifier of A and B .

Exercises: Try to prove the following, using the above rules.

1. $(\exists x)(\forall y)R(x, y) \supset (\forall y)(\exists x)R(x, y)$
2. $(\exists x)[P(x) \supset (\forall x)P(x)]$
3. $(\forall x)(\forall y)[P(x) \wedge P(y)] \supset (\exists x)(\exists y)[P(x) \wedge P(y)]$
4. $(\forall x)(\forall y)[P(x) \wedge P(y)] \supset (\forall x)(\forall y)[P(x) \wedge P(y)]$
5. $(\forall x)(\exists y)(\forall z)(\exists w)[R(x, y) \vee \neg R(w, z)]$
6. $(\exists x)(\forall y)[P(y) \uparrow (P(x) \uparrow Q(x))] \supset (\forall x)Q(x)$

MGU stands for most general unifier.

Definition — MGU Atomic Closure Rule

199

Suppose T is a tableau for a set S of sentences of \mathbf{L} and that some branch of T contains atomic literals A and $\neg B$. Then $T\sigma$ is also a tableau for S , where σ is the most general unifier of A and B .

Exercises: Try to prove the following, using the above rules.

1. $(\exists x)(\forall y)R(x, y) \supset (\forall y)(\exists x)R(x, y)$
2. $(\exists x)[P(x) \supset (\forall x)P(x)]$
3. $(\forall x)(\forall y)[P(x) \wedge P(y)] \supset (\exists x)(\exists y)[P(x) \wedge P(y)]$
4. $(\forall x)(\forall y)[P(x) \wedge P(y)] \supset (\forall x)(\forall y)[P(x) \wedge P(y)]$
5. $(\forall x)(\exists y)(\forall z)(\exists w)[R(x, y) \vee \neg R(w, z)]$
6. $(\exists x)(\forall y)[P(y) \uparrow (P(x) \uparrow Q(x))] \supset (\forall x)Q(x)$

Definition — Free-Variable Resolution Expansion Rules 200

$$\frac{\gamma}{\gamma(x)} \text{ (} x \text{ new variable)}$$

$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (} f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta)$$

Definition — Substitution is Free for R

201

Let σ be a substitution and R be a resolution expansion. We extend σ to R by setting $R\sigma$ to be the result of replacing each formula X in R by $X\sigma$. We say σ is *free for R* , provided that σ is free for every formula in R .

Definition — Resolution Substitution Rule

202

If R is a resolution expansion for a set S of sentences of \mathbf{L} and if the substitution σ is free for R , then $R\sigma$ is also a resolution

Definition — Free-Variable Resolution Expansion Rules 200

$$\frac{\gamma}{\gamma(x)} \text{ (}x \text{ new variable)}$$
$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (}f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta\text{)}$$

Definition — Substitution is Free for R 201

Let σ be a substitution and R be a resolution expansion. We extend σ to R by setting $R\sigma$ to be the result of replacing each formula X in R by $X\sigma$. We say σ is *free for R* , provided that σ is free for every formula in R .

Definition — Resolution Substitution Rule 202

If R is a resolution expansion for a set S of sentences of \mathbf{L} and if the substitution σ is free for R , then $R\sigma$ is also a resolution

Definition — Free-Variable Resolution Expansion Rules 200

$$\frac{\gamma}{\gamma(x)} \text{ (}x \text{ new variable)}$$
$$\frac{\delta}{\delta(f(x_1, \dots, x_n))} \text{ (}f \text{ new Skolem function symbol and } x_1, \dots, x_n \text{ free variables of } \delta\text{)}$$

Definition — Substitution is Free for R 201

Let σ be a substitution and R be a resolution expansion. We extend σ to R by setting $R\sigma$ to be the result of replacing each formula X in R by $X\sigma$. We say σ is *free for R* , provided that σ is free for every formula in R .

Definition — Resolution Substitution Rule 202

If R is a resolution expansion for a set S of sentences of \mathbf{L} and if the substitution σ is free for R , then $R\sigma$ is also a resolution expansion for S .

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
 2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ $\alpha_1 [1.]$
 3. $\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ $\alpha_2 [1.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ $\alpha_1 [1.]$
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ $\alpha_2 [1.]$
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a) [2.]$
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1) [3.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1)$ [3.]
6. $[\neg(\exists y)R(b(v_1), v_1, y)]$ $\delta(b(v_1))$ [5.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ $\alpha_1 [1.]$
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ $\alpha_2 [1.]$
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a) [2.]$
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1) [3.]$
6. $[\neg(\exists y)R(b(v_1), v_1, y)]$ $\delta(b(v_1)) [5.]$
7. $[R(v_2, a, f(v_2, a))]$ $\gamma(v_2) [4.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ $\alpha_1 [1.]$
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ $\alpha_2 [1.]$
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a) [2.]$
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1) [3.]$
6. $[\neg(\exists y)R(b(v_1), v_1, y)]$ $\delta(b(v_1)) [5.]$
7. $[R(v_2, a, f(v_2, a))]$ $\gamma(v_2) [4.]$
8. $[\neg R(b(v_1), v_1, v_3)]$ $\gamma(v_3) [6.]$

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1)$ [3.]
6. $[\neg(\exists y)R(b(v_1), v_1, y)]$ $\delta(b(v_1))$ [5.]
7. $[R(v_2, a, f(v_2, a))]$ $\gamma(v_2)$ [4.]
8. $[\neg R(b(v_1), v_1, v_3)]$ $\gamma(v_3)$ [6.]

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w))] \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]
5. $[\neg(\forall x)(\exists y)R(x, v_1, y)]$ $\gamma(v_1)$ [3.]
6. $[\neg(\exists y)R(b(v_1), v_1, y)]$ $\delta(b(v_1))$ [5.]
7. $[R(v_2, a, f(v_2, a))]$ $\gamma(v_2)$ [4.]
8. $[\neg R(b(v_1), v_1, v_3)]$ $\gamma(v_3)$ [6.]

Res.Subst.Rule: $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ (**free for R**)

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ $\alpha_1 [1.]$
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ $\alpha_2 [1.]$
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a) [2.]$
- 5'. $[\neg(\forall x)(\exists y)R(x, a, y)]$ $\gamma(a) [3.]$
- 6'. $[\neg(\exists y)R(b(a), a, y)]$ $\delta(b(a)) [5.]$
- 7'. $[R(b(a), a, f(b(a), a))]$ $\gamma(b(a)) [4.]$
- 8'. $[\neg R(b(a), a, f(b(a), a))]$ $\gamma(f(b(a), a)) [6.]$

Res.Subst.Rule: $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ (**free for R**)

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w))] \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]
- 5'. $[\neg(\forall x)(\exists y)R(x, a, y)]$ $\gamma(a)$ [3.]
- 6'. $[\neg(\exists y)R(b(a), a, y)]$ $\delta(b(a))$ [5.]
- 7'. $[R(b(a), a, f(b(a), a))]$ $\gamma(b(a))$ [4.]
- 8'. $[\neg R(b(a), a, f(b(a), a))]$ $\gamma(f(b(a), a))$ [6.]

Res.Subst.Rule: $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ (**free for R**)

9. $[]$ *resolve [7., 8.]*

Example — $(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)$ 203

1. $[\neg(\exists w)(\forall x)R(x, w, f(x, w)) \supset (\exists w)(\forall x)(\exists y)R(x, w, y)]$
2. $[(\exists w)(\forall x)R(x, w, f(x, w))]$ α_1 [1.]
3. $[\neg(\exists w)(\forall x)(\exists y)R(x, w, y)]$ α_2 [1.]
4. $[(\forall x)R(x, a, f(x, a))]$ $\delta(a)$ [2.]
- 5'. $[\neg(\forall x)(\exists y)R(x, a, y)]$ $\gamma(a)$ [3.]
- 6'. $[\neg(\exists y)R(b(a), a, y)]$ $\delta(b(a))$ [5.]
- 7'. $[R(b(a), a, f(b(a), a))]$ $\gamma(b(a))$ [4.]
- 8'. $[\neg R(b(a), a, f(b(a), a))]$ $\gamma(f(b(a), a))$ [6.]

Res.Subst.Rule: $\sigma = \{v_1/a, v_2/b(a), v_3/f(b(a), a)\}$ (**free for R**)

9. [] resolve [7., 8.]

As before:

How can the substitution σ be determined? — use unification

When/where to apply unification? — restrict to atomic literals

This also avoids expensive freeness checks. — Can you see why?

Definition — Binary Resolution Rule

204

Suppose R is a resolution expansion for the set S of sentences from \mathbf{L} , R contains the generalized disjunctions

$$[X, Y_1, \dots, Y_m] \quad \text{and} \quad [\neg Z, W_1, \dots, W_p]$$

where X and Z are atomic. Moreover, let σ be a most general unifier of X and Z .

Then $R^*\sigma$ is also a resolution expansion for S , where R^* is R with

$$[Y_1, \dots, Y_m, W_1, \dots, W_p]$$

added.

(Note: $R^*\sigma$ is far too strong in practice)

Usual Presentation (the Y_i and W_i are typically literals) :

$$\frac{[X, Y_1, \dots, Y_m] \quad [\neg Z, W_1, \dots, W_p] \quad \sigma \text{ MGU of atoms } X \text{ and } Z}{[Y_1, \dots, Y_m, W_1, \dots, W_p]\sigma} \text{ resolve}$$

Definition — Binary Resolution Rule

204

Suppose R is a resolution expansion for the set S of sentences from \mathbf{L} , R contains the generalized disjunctions

$$[X, Y_1, \dots, Y_m] \quad \text{and} \quad [\neg Z, W_1, \dots, W_p]$$

where X and Z are atomic. Moreover, let σ be a most general unifier of X and Z .

Then $R^*\sigma$ is also a resolution expansion for S , where R^* is R with

$$[Y_1, \dots, Y_m, W_1, \dots, W_p]$$

added.

(Note: $R^*\sigma$ is far too strong in practice)

Usual Presentation (the Y_i and W_i are typically literals) :

$$\frac{[X, Y_1, \dots, Y_m] \quad [\neg Z, W_1, \dots, W_p] \quad \sigma \text{ MGU of atoms } X \text{ and } Z}{[Y_1, \dots, Y_m, W_1, \dots, W_p]\sigma} \text{ resolve}$$

Definition — Binary Resolution Rule

204

Suppose R is a resolution expansion for the set S of sentences from \mathbf{L} , R contains the generalized disjunctions

$$[X, Y_1, \dots, Y_m] \quad \text{and} \quad [\neg Z, W_1, \dots, W_p]$$

where X and Z are atomic. Moreover, let σ be a most general unifier of X and Z .

Then $R^*\sigma$ is also a resolution expansion for S , where R^* is R with

$$[Y_1, \dots, Y_m, W_1, \dots, W_p]$$

added. (Note: $R^*\sigma$ is far too strong in practice)

Usual Presentation (the Y_i and W_i are typically literals) :

$$\frac{[X, Y_1, \dots, Y_m] \quad [\neg Z, W_1, \dots, W_p] \quad \sigma \text{ MGU of atoms } X \text{ and } Z}{[Y_1, \dots, Y_m, W_1, \dots, W_p]\sigma} \text{ resolve}$$

Problem: System is not yet complete! . . . see blackboard . . .

$$(\forall x)[P(x) \vee P(x)] \supset (\exists y)[P(y) \wedge P(y)]$$

. . .
[$P(x), P(x)$]
[$\neg P(y), \neg P(y)$]
. . .

Problem: System is not yet complete! . . . see blackboard . . .

$$(\forall x)[P(x) \vee P(x)] \supset (\exists y)[P(y) \wedge P(y)]$$

. . .

$$[P(x), P(x)]$$

$$[\neg P(y), \neg P(y)]$$

. . .

Definition — Factoring Rule

Suppose R is a resolution expansion for the set S of sentences of \mathbf{L} , R contains the generalized disjunction

$$[X, Y, Z_1, \dots, Z_n]$$

where X and Y are literals, and σ is most general unifier of X and Y . Then $R^*\sigma$ is also a resolution expansion for S , where R^* is R with

$$[Y, Z_1, \dots, Z_n]$$

added. (Note: $R^*\sigma$ is far too strong in practice)

Usual Presentation (the Z_i are typically literals):

$$\frac{[X, Y, Z_1, \dots, Z_n] \quad \sigma \text{ MGU of literals } X \text{ and } Y}{[Y, Z_1, \dots, Z_n]\sigma} \text{ factoring}$$

Definition — Factoring Rule

Suppose R is a resolution expansion for the set S of sentences of \mathbf{L} , R contains the generalized disjunction

$$[X, Y, Z_1, \dots, Z_n]$$

where X and Y are literals, and σ is most general unifier of X and Y . Then $R^*\sigma$ is also a resolution expansion for S , where R^* is R with

$$[Y, Z_1, \dots, Z_n]$$

added. (Note: $R^*\sigma$ is far too strong in practice)

Usual Presentation (the Z_i are typically literals):

$$\frac{[X, Y, Z_1, \dots, Z_n] \quad \sigma \text{ MGU of literals } X \text{ and } Y}{[Y, Z_1, \dots, Z_n]\sigma} \text{ factoring}$$

Theorem — Soundness

206

1. If the sentence X has a free-variable tableau proof, then X is valid.
2. If the sentence X has a free-variable resolution proof, then X is valid.

Proof: ... similar to before ...

Theorem — Completeness (without restrictions)

207

1. If X is valid, then X has a free-variable tableau proof.
2. If X is valid, then X has a free-variable resolution proof.

Proof: ... not here; Abstract Consistency Method can be used ...

Theorem — Soundness

206

1. If the sentence X has a free-variable tableau proof, then X is valid.
2. If the sentence X has a free-variable resolution proof, then X is valid.

Proof: ... similar to before ...**Theorem — Completeness (without restrictions)**

207

1. If X is valid, then X has a free-variable tableau proof.
2. If X is valid, then X has a free-variable resolution proof.

Proof: ... not here; Abstract Consistency Method can be used ...



- ▶ In order to implement the tableau procedure we want to introduce further restrictions to our rules while still maintaining completeness.
- ▶ Below some possible restrictions are presented; they preserve completeness.
- ▶ Many other strategies and heuristics are possible and have been applied in systems.

Definition — Most General Atomic Closure Substitution 208

Suppose T is a tableau with branches $\Theta_0, \Theta_1, \dots, \Theta_n$ and for each i let A_i and $\neg B_i$ be a pair of atomic literals on branch Θ_i . If σ is a most general solution of the family of equations

$$A_0 = B_0, A_1 = B_1, \dots, A_n = B_n$$

then we call σ a *most general atomic closing substitution*.

- ▶ In order to implement the tableau procedure we want to introduce further restrictions to our rules while still maintaining completeness.
- ▶ Below some possible restrictions are presented; they preserve completeness.
- ▶ Many other strategies and heuristics are possible and have been applied in systems.

Definition — Most General Atomic Closure Substitution 208

Suppose T is a tableau with branches $\Theta_0, \Theta_1, \dots, \Theta_n$ and for each i let A_i and $\neg B_i$ be a pair of atomic literals on branch Θ_i . If σ is a most general solution of the family of equations

$$A_0 = B_0, A_1 = B_1, \dots, A_n = B_n$$

then we call σ a *most general atomic closing substitution*.

Lemma — Lifting Lemma

209

Suppose T be a tableau, and τ be a substitution, free for T , such that each branch of $T\sigma$ is atomically closed. Then there is a most general atomic closure substitution σ for T .

Proof: ... not here ...

- ▶ Thus, with this restriction we look only for simultaneous atomic closure in all branches of a tableau.
- ▶ Note that completeness of this restriction does not follow 'for free' from the Tableau Substitution Rule.

Lemma — Lifting Lemma

209

Suppose T be a tableau, and τ be a substitution, free for T , such that each branch of $T\sigma$ is atomically closed. Then there is a most general atomic closure substitution σ for T .

Proof: ... not here ...

- ▶ Thus, with this restriction we look only for simultaneous atomic closure in all branches of a tableau.
- ▶ Note that completeness of this restriction does not follow 'for free' from the Tableau Substitution Rule.

Lemma — Lifting Lemma

209

Suppose T be a tableau, and τ be a substitution, free for T , such that each branch of $T\sigma$ is atomically closed. Then there is a most general atomic closure substitution σ for T .

Proof: ... not here ...

- ▶ Thus, with this restriction we look only for simultaneous atomic closure in all branches of a tableau.
- ▶ Note that completeness of this restriction does not follow 'for free' from the Tableau Substitution Rule.

Definition — Tableau Construction Rule

210

A *tableau construction rule* is a rule R that, when supplied with a tableau T and some side information either says

1. no continuation is possible, or
2. produces a new tableau T' and new side information,

where T' results from T by application of a single Tableau Expansion Rule to T .

We also assume that R specifies outright the appropriate side information to accompany the initial tableau of an attempt to prove the sentence X .

Definition — Tableau Construction Rule

210

A *tableau construction rule* is a rule R that, when supplied with a tableau T and some side information either says

1. no continuation is possible, or
2. produces a new tableau T' and new side information,

where T' results from T by application of a single Tableau Expansion Rule to T .

We also assume that R specifies outright the appropriate side information to accompany the initial tableau of an attempt to prove the sentence X .

Example — Tableau Construction Rule R

211

- ▶ Side information maintained by R :
 - ▶ which formula occurrences have been used on which branches
 - ▶ a priority ordering for formula occurrences on each branch
 - ▶ a priority ordering for branches
 - ▶ initial side condition: sentence $\neg X$ has not been used yet, it has top priority on its branch; the single branch has top priority
- ▶ Now, R does the following:
 - ▶ if every non-literal formula occurrence has been used on every branch, then no further tableau construction is possible
 - ▶ otherwise
 - ▶ select branch Θ with highest priority with unused non-literal
 - ▶ select unused non-literal Z on Θ with highest priority
 - ▶ apply an tableau expansion rule to Z on Θ , producing T'
 - ▶ add side information to T' : occurrence of Z on Θ has been used; given the extended branches on T' the lowest priority; on these branches give the added formulas the lowest priority
 - ▶ note: R treats γ -formulas like all others; no reuse possible(!)

Example — Tableau Construction Rule R

211

- ▶ Side information maintained by R :
 - ▶ which formula occurrences have been used on which branches
 - ▶ a priority ordering for formula occurrences on each branch
 - ▶ a priority ordering for branches
 - ▶ initial side condition: sentence $\neg X$ has not been used yet, it has top priority on its branch; the single branch has top priority
- ▶ Now, R does the following:
 - ▶ if every non-literal formula occurrence has been used on every branch, then no further tableau construction is possible
 - ▶ otherwise
 - ▶ select branch Θ with highest priority with unused non-literal
 - ▶ select unused non-literal Z on Θ with highest priority
 - ▶ apply an tableau expansion rule to Z on Θ , producing T'
 - ▶ add side information to T' : occurrence of Z on Θ has been used; given the extended branches on T' the lowest priority; on these branches give the added formulas the lowest priority
 - ▶ note: R treats γ -formulas like all others; no reuse possible(!)

Definition — Sequence for Φ according to rule R

212

Let Φ be a sentence of L . We say a sequence of tableaux (and associated side information) T_1, T_2, T_3, \dots (finite or infinite) is a *sequence for Φ constructed according to rule R* provided T_1 is the tableau with one branch, that branch containing only Φ , and with the side information associated with it by R , and provided each T_{i+1} and its side information result from T_i and its side information by an application of rule R .

Definition — Fairness (Tableaux)

213

A tableau construction rule is *fair* provided, for any sentence Φ , the sequence T_1, T_2, \dots of tableaux for Φ constructed according to R has the following properties (for each n):

1. Every non-literal formula occurrence in T_n eventually has the appropriate Tableau Expansion Rule applied to it, on each branch on which it occurs.
2. Every γ -formula occurrence in T_n has the γ -rule applied to it arbitrarily often, on each branch on which it occurs.

Note: Rule R from the example above is not fair.

Definition — Fairness (Tableaux)

213

A tableau construction rule is *fair* provided, for any sentence Φ , the sequence T_1, T_2, \dots of tableaux for Φ constructed according to R has the following properties (for each n):

1. Every non-literal formula occurrence in T_n eventually has the appropriate Tableau Expansion Rule applied to it, on each branch on which it occurs.
2. Every γ -formula occurrence in T_n has the γ -rule applied to it arbitrarily often, on each branch on which it occurs.

Note: Rule R from the example above is not fair.

Theorem — Completeness (Tableaux with Restrictions) 214

Let R be any fair tableau construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Tableau Expansion Rule applications come first and are according to rule R .
2. A single Tableau Substitution Rule application follows, using a substitution σ that is a most general atomic closure substitution.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

Theorem — Completeness (Tableaux with Restrictions) 214

Let R be any fair tableau construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Tableau Expansion Rule applications come first and are according to rule R .
2. A single Tableau Substitution Rule application follows, using a substitution σ that is a most general atomic closure substitution.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

Theorem — Completeness (Tableaux with Restrictions) 214

Let R be any fair tableau construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Tableau Expansion Rule applications come first and are according to rule R .
2. A single Tableau Substitution Rule application follows, using a substitution σ that is a most general atomic closure substitution.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

- ▶ Resolution construction rules are analogous.

Definition — Fairness (Resolution)

215

A resolution construction rule R is *fair*, provided, for any sentence Φ , the sequence R_1, R_2, R_3, \dots of resolution expansions for Φ constructed according to R has the following properties (for each n):

1. Every disjunction in R_n containing a non-literal eventually has some resolution expansion applied to it.
2. If the γ -rule is applied to a disjunction in R_n , then it is applied to that disjunction arbitrarily often.

- ▶ Resolution construction rules are analogous.

Definition — Fairness (Resolution)

215

A resolution construction rule R is *fair*, provided, for any sentence Φ , the sequence R_1, R_2, R_3, \dots of resolution expansions for Φ constructed according to R has the following properties (for each n):

1. Every disjunction in R_n containing a non-literal eventually has some resolution expansion applied to it.
2. If the γ -rule is applied to a disjunction in R_n , then it is applied to that disjunction arbitrarily often.

Theorem — Completeness (Resolution with Restrictions) 216

Let R be any fair resolution construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Resolution Expansion Rule applications come first and are according to rule R .
2. These are followed by applications of the Binary Resolution Rule and the Factorization rule, to proper clauses only.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

Theorem — Completeness (Resolution with Restrictions) 216

Let R be any fair resolution construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Resolution Expansion Rule applications come first and are according to rule R .
2. These are followed by applications of the Binary Resolution Rule and the Factorization rule, to proper clauses only.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

Theorem — Completeness (Resolution with Restrictions) 216

Let R be any fair resolution construction rule. If X is a valid sentence of L , X has a proof in which the following apply:

1. All Resolution Expansion Rule applications come first and are according to rule R .
2. These are followed by applications of the Binary Resolution Rule and the Factorization rule, to proper clauses only.

Proof: ... not here ...

Exercises: Formulate (and implement?) a fair resolution construction rule.

- ▶ It is often useful to rewrite a sentence before trying to prove it.
- ▶ Such a rewriting may simplify the problem for the prover and ease proof search.
- ▶ We therefore establish a Replacement Theorem

Theorem — Replacement Theorem

217

Let $\Phi(A)$, X , and Y be first-order formulas of the language \mathbf{L} . Let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \equiv Y$ is true in \mathbf{M} , then $\Phi(X) \equiv \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Corollary

218

If $X \equiv Y$ is valid, then $\Phi(X) \equiv \Phi(Y)$ is valid.

- ▶ It is often useful to rewrite a sentence before trying to prove it.
- ▶ Such a rewriting may simplify the problem for the prover and ease proof search.
- ▶ We therefore establish a Replacement Theorem

Theorem — Replacement Theorem

217

Let $\Phi(A)$, X , and Y be first-order formulas of the language \mathbf{L} . Let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \equiv Y$ is true in \mathbf{M} , then $\Phi(X) \equiv \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Corollary

218

If $X \equiv Y$ is valid, then $\Phi(X) \equiv \Phi(Y)$ is valid.

- ▶ It is often useful to rewrite a sentence before trying to prove it.
- ▶ Such a rewriting may simplify the problem for the prover and ease proof search.
- ▶ We therefore establish a Replacement Theorem

Theorem — Replacement Theorem

217

Let $\Phi(A)$, X , and Y be first-order formulas of the language \mathbf{L} . Let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \equiv Y$ is true in \mathbf{M} , then $\Phi(X) \equiv \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Corollary

218

If $X \equiv Y$ is valid, then $\Phi(X) \equiv \Phi(Y)$ is valid.

- ▶ It is often useful to rewrite a sentence before trying to prove it.
- ▶ Such a rewriting may simplify the problem for the prover and ease proof search.
- ▶ We therefore establish a Replacement Theorem

Theorem — Replacement Theorem

217

Let $\Phi(A)$, X , and Y be first-order formulas of the language \mathbf{L} . Let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \equiv Y$ is true in \mathbf{M} , then $\Phi(X) \equiv \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Corollary

218

If $X \equiv Y$ is valid, then $\Phi(X) \equiv \Phi(Y)$ is valid.

- ▶ We may also want to 'kind of rewrite' with implications.
- ▶ Therefore we introduce a stronger version of replacement
- ▶ where only formulas that occur 'positively' can be replaced.

Definition — Positive Formula Occurrences

219

We say that all occurrences of A in $\Phi(A)$ are *positive* provided:

1. $\Phi(A) = A$.
2. $\Phi(A) = \neg\neg\Psi(A)$ and all occurrences of A in $\Psi(A)$ are positive.
3. $\Phi(A) = \alpha$ and the occurrences of A in α_1 and α_2 are positive.
4. $\Phi(A) = \beta$ and the occurrences of A in β_1 and β_2 are positive.
5. $\Phi(A) = \gamma$ (with x being the quantified variable) and all occurrences of A in $\gamma(x)$ are positive.
6. $\Phi(A) = \delta$ (with x being the quantified variable) and all occurrences of A in $\delta(x)$ are positive.

- ▶ We may also want to 'kind of rewrite' with implications.
- ▶ Therefore we introduce a stronger version of replacement
- ▶ where only formulas that occur 'positively' can be replaced.

Definition — Positive Formula Occurrences

219

We say that all occurrences of A in $\Phi(A)$ are *positive* provided:

1. $\Phi(A) = A$.
2. $\Phi(A) = \neg\neg\Psi(A)$ and all occurrences of A in $\Psi(A)$ are positive.
3. $\Phi(A) = \alpha$ and the occurrences of A in α_1 and α_2 are positive.
4. $\Phi(A) = \beta$ and the occurrences of A in β_1 and β_2 are positive.
5. $\Phi(A) = \gamma$ (with x being the quantified variable) and all occurrences of A in $\gamma(x)$ are positive.
6. $\Phi(A) = \delta$ (with x being the quantified variable) and all occurrences of A in $\delta(x)$ are positive.

Example — $(\forall x)[P(x, y) \supset \neg(\exists y)\neg R(x, y)]$

220

... discuss on blackboard ...

Exercise: ... maybe implement this function ...

Theorem — Implicational Replacement Theorem

221

Let $\Phi(A)$ be a formula in which the atomic formula A has only positive occurrences. Let X and Y be first-order formulas of the language L , and let $M = \langle D, I \rangle$ be a model for L .

If $X \supset Y$ is true in M , then $\Phi(X) \supset \Phi(Y)$ is also true in M .

Proof: ... not here ...

Example — $(\forall x)[P(x, y) \supset \neg(\exists y)\neg R(x, y)]$

220

... discuss on blackboard ...

Exercise: ... maybe implement this function ...

Theorem — Implicational Replacement Theorem

221

Let $\Phi(A)$ be a formula in which the atomic formula A has only positive occurrences. Let X and Y be first-order formulas of the language L , and let $M = \langle D, I \rangle$ be a model for L .

If $X \supset Y$ is true in M , then $\Phi(X) \supset \Phi(Y)$ is also true in M .

Proof: ... not here ...

Example — $(\forall x)[P(x, y) \supset \neg(\exists y)\neg R(x, y)]$

220

... discuss on blackboard ...

Exercise: ... maybe implement this function ...

Theorem — Implicational Replacement Theorem

221

Let $\Phi(A)$ be a formula in which the atomic formula A has only positive occurrences. Let X and Y be first-order formulas of the language \mathbf{L} , and let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \supset Y$ is true in \mathbf{M} , then $\Phi(X) \supset \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Example — $(\forall x)[P(x, y) \supset \neg(\exists y)\neg R(x, y)]$

220

... discuss on blackboard ...

Exercise: ... maybe implement this function ...

Theorem — Implicational Replacement Theorem

221

Let $\Phi(A)$ be a formula in which the atomic formula A has only positive occurrences. Let X and Y be first-order formulas of the language \mathbf{L} , and let $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ be a model for \mathbf{L} .

If $X \supset Y$ is true in \mathbf{M} , then $\Phi(X) \supset \Phi(Y)$ is also true in \mathbf{M} .

Proof: ... not here ...

Definition — Negative Occurrences

222

A formula A has only *negative* occurrences in $\Phi(A)$ provided A has only positive occurrences in $\neg\Phi(A)$.

Corollary

223

Suppose all occurrences of formula A in $\Phi(A)$ are negative. If $Y \supset X$ is true in the model \mathbf{M} , so is $\Phi(X) \supset \Phi(Y)$.

Definition — Negative Occurrences

222

A formula A has only *negative* occurrences in $\Phi(A)$ provided A has only positive occurrences in $\neg\Phi(A)$.

Corollary

223

Suppose all occurrences of formula A in $\Phi(A)$ are negative. If $Y \supset X$ is true in the model \mathbf{M} , so is $\Phi(X) \supset \Phi(Y)$.

- ▶ In free-variable tableaux and resolution (Skolem) functions were introduced with each application of the δ -rule.
- ▶ It is possible to preprocess the sentences in problem and to introduce these Skolem functions a priori.
- ▶ δ -rule application are not needed then anymore.
- ▶ The proof search may then become more efficient.

Lemma

224

Let Ψ be a formula with free variables among x, y_1, \dots, y_n and let f be an n -place function symbol that does not occur in Ψ . If $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ is any model, there are models $\mathbf{M}_1 = \langle \mathbf{D}, \mathbf{J}_1 \rangle$ and $\mathbf{M}_2 = \langle \mathbf{D}, \mathbf{J}_2 \rangle$ where \mathbf{I} , \mathbf{J}_1 , and \mathbf{J}_2 differ only in the interpretation of f , and

1. In \mathbf{M}_1 , $(\exists x)\Psi \supset \Psi\{x/f(y_1, \dots, y_n)\}$ is true.
2. In \mathbf{M}_2 , $\Psi\{x/f(y_1, \dots, y_n)\} \supset (\forall x)\Psi$ is true.

Proof: ... not here, maybe exercise ...

- ▶ In free-variable tableaux and resolution (Skolem) functions were introduced with each application of the δ -rule.
- ▶ It is possible to preprocess the sentences in problem and to introduce these Skolem functions a priori.
- ▶ δ -rule application are not needed then anymore.
- ▶ The proof search may then become more efficient.

Lemma

224

Let Ψ be a formula with free variables among x, y_1, \dots, y_n and let f be an n -place function symbol that does not occur in Ψ . If $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ is any model, there are models $\mathbf{M}_1 = \langle \mathbf{D}, \mathbf{J}_1 \rangle$ and $\mathbf{M}_2 = \langle \mathbf{D}, \mathbf{J}_2 \rangle$ where \mathbf{I} , \mathbf{J}_1 , and \mathbf{J}_2 differ only in the interpretation of f , and

1. In \mathbf{M}_1 , $(\exists x)\Psi \supset \Psi\{x/f(y_1, \dots, y_n)\}$ is true.
2. In \mathbf{M}_2 , $\Psi\{x/f(y_1, \dots, y_n)\} \supset (\forall x)\Psi$ is true.

Proof: ... not here, maybe exercise ...

- ▶ In free-variable tableaux and resolution (Skolem) functions were introduced with each application of the δ -rule.
- ▶ It is possible to preprocess the sentences in problem and to introduce these Skolem functions a priori.
- ▶ δ -rule application are not needed then anymore.
- ▶ The proof search may then become more efficient.

Lemma

224

Let Ψ be a formula with free variables among x, y_1, \dots, y_n and let f be an n -place function symbol that does not occur in Ψ . If $\mathbf{M} = \langle \mathbf{D}, \mathbf{I} \rangle$ is any model, there are models $\mathbf{M}_1 = \langle \mathbf{D}, \mathbf{J}_1 \rangle$ and $\mathbf{M}_2 = \langle \mathbf{D}, \mathbf{J}_2 \rangle$ where \mathbf{I} , \mathbf{J}_1 , and \mathbf{J}_2 differ only in the interpretation of f , and

1. In \mathbf{M}_1 , $(\exists x)\Psi \supset \Psi\{x/f(y_1, \dots, y_n)\}$ is true.
2. In \mathbf{M}_2 , $\Psi\{x/f(y_1, \dots, y_n)\} \supset (\forall x)\Psi$ is true.

Proof: ... not here, maybe exercise ...

Theorem — Skolemization

225

Let $\Psi(x)$ be a formula with free variables x, y_1, \dots, y_n , and let $\Phi(A)$ be a formula such that $\Phi((\exists x)\Psi(x))$ is a sentence. Finally suppose that f is a (Skolem) function symbol that does not occur in $\Phi((\exists x)\Psi(x))$.

1. If all occurrences of A are positive in $\Phi(A)$, then $\Phi((\exists x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.
2. If all occurrences of A are negative in $\Phi(A)$, then $\Phi((\forall x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.

Proof: ... not here, maybe exercise ...

Slogan: 'Skolemization does not preserve models, but it preserves satisfiability.'

(This property is all we need in refutation based theorem proving.)

Theorem — Skolemization

225

Let $\Psi(x)$ be a formula with free variables x, y_1, \dots, y_n , and let $\Phi(A)$ be a formula such that $\Phi((\exists x)\Psi(x))$ is a sentence. Finally suppose that f is a (Skolem) function symbol that does not occur in $\Phi((\exists x)\Psi(x))$.

1. If all occurrences of A are positive in $\Phi(A)$, then $\Phi((\exists x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.
2. If all occurrences of A are negative in $\Phi(A)$, then $\Phi((\forall x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.

Proof: ... not here, maybe exercise ...

Slogan: 'Skolemization does not preserve models, but it preserves satisfiability.'

(This property is all we need in refutation based theorem proving.)

Theorem — Skolemization

225

Let $\Psi(x)$ be a formula with free variables x, y_1, \dots, y_n , and let $\Phi(A)$ be a formula such that $\Phi((\exists x)\Psi(x))$ is a sentence. Finally suppose that f is a (Skolem) function symbol that does not occur in $\Phi((\exists x)\Psi(x))$.

1. If all occurrences of A are positive in $\Phi(A)$, then $\Phi((\exists x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.
2. If all occurrences of A are negative in $\Phi(A)$, then $\Phi((\forall x)\Psi(x))$ is satisfiable if and only if $\Phi(\Psi(f(y_1, \dots, y_n)))$ is satisfiable.

Proof: ... not here, maybe exercise ...

Slogan: 'Skolemization does not preserve models, but it preserves satisfiability.'

(This property is all we need in refutation based theorem proving.)

Example — Skolemization

226

- ▶ $X_1 := (\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \supset (\exists w)P(w)]$
the subformula beginning with $(\exists y)$ occurs positively
- ▶ $X_2 := (\forall x)[(\exists z)(\forall w)R(x, f(x), z, w) \supset (\exists w)P(w)]$
Skolemization theorem: X_1 is satisfiable if and only if X_2 is.
The subformula of X_2 beginning with $(\forall w)$ occurs negatively.
- ▶ $X_3 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset (\exists w)P(w)]$
Skolemization theorem: X_2 is satisfiable if and only if X_3 is.
The subformula of X_3 beginning with $(\exists w)$ occurs negatively.
- ▶ $X_4 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset P(a)]$ Skolemization theorem: X_3 is satisfiable if and only if X_4 is.
- ▶ Process stops since there are no more positive existentials or negative universals to replace.

Example — Skolemization

226

- ▶ $X_1 := (\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \supset (\exists w)P(w)]$
the subformula beginning with $(\exists y)$ occurs positively
- ▶ $X_2 := (\forall x)[(\exists z)(\forall w)R(x, f(x), z, w) \supset (\exists w)P(w)]$
Skolemization theorem: X_1 is satisfiable if and only if X_2 is.
The subformula of X_2 beginning with $(\forall w)$ occurs negatively.
- ▶ $X_3 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset (\exists w)P(w)]$
Skolemization theorem: X_2 is satisfiable if and only if X_3 is.
The subformula of X_3 beginning with $(\exists w)$ occurs negatively.
- ▶ $X_4 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset P(a)]$ Skolemization theorem: X_3 is satisfiable if and only if X_4 is.
- ▶ Process stops since there are no more positive existentials or negative universals to replace.

Example — Skolemization

226

- ▶ $X_1 := (\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \supset (\exists w)P(w)]$
the subformula beginning with $(\exists y)$ occurs positively
- ▶ $X_2 := (\forall x)[(\exists z)(\forall w)R(x, f(x), z, w) \supset (\exists w)P(w)]$
Skolemization theorem: X_1 is satisfiable if and only if X_2 is.
The subformula of X_2 beginning with $(\forall w)$ occurs negatively.
- ▶ $X_3 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset (\exists w)P(w)]$
Skolemization theorem: X_2 is satisfiable if and only if X_3 is.
The subformula of X_3 beginning with $(\exists w)$ occurs negatively.
- ▶ $X_4 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset P(a)]$ Skolemization theorem: X_3 is satisfiable if and only if X_4 is.
- ▶ Process stops since there are no more positive existentials or negative universals to replace.

Example — Skolemization

226

- ▶ $X_1 := (\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \supset (\exists w)P(w)]$
the subformula beginning with $(\exists y)$ occurs positively
- ▶ $X_2 := (\forall x)[(\exists z)(\forall w)R(x, f(x), z, w) \supset (\exists w)P(w)]$
Skolemization theorem: X_1 is satisfiable if and only if X_2 is.
The subformula of X_2 beginning with $(\forall w)$ occurs negatively.
- ▶ $X_3 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset (\exists w)P(w)]$
Skolemization theorem: X_2 is satisfiable if and only if X_3 is.
The subformula of X_3 beginning with $(\exists w)$ occurs negatively.
- ▶ $X_4 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset P(a)]$ Skolemization theorem: X_3 is satisfiable if and only if X_4 is.
- ▶ Process stops since there are no more positive existentials or negative universals to replace.

Example — Skolemization

226

- ▶ $X_1 := (\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \supset (\exists w)P(w)]$
the subformula beginning with $(\exists y)$ occurs positively
- ▶ $X_2 := (\forall x)[(\exists z)(\forall w)R(x, f(x), z, w) \supset (\exists w)P(w)]$
Skolemization theorem: X_1 is satisfiable if and only if X_2 is.
The subformula of X_2 beginning with $(\forall w)$ occurs negatively.
- ▶ $X_3 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset (\exists w)P(w)]$
Skolemization theorem: X_2 is satisfiable if and only if X_3 is.
The subformula of X_3 beginning with $(\exists w)$ occurs negatively.
- ▶ $X_4 := (\forall x)[(\exists z)R(x, f(x), z, g(x, z)) \supset P(a)]$ Skolemization theorem: X_3 is satisfiable if and only if X_4 is.
- ▶ Process stops since there are no more positive existentials or negative universals to replace.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Example — Skolemization – Work outside to inside

227

$$(\forall x)(\exists y)(\exists z)R(x, y, z)$$

From inside to outside

$$(\forall x)(\exists y)R(x, y, f(x, y))$$

$$(\forall x)R(x, k(x), f(x, k(x)))$$

From outside to inside

$$(\forall x)(\exists z)R(x, k(x), z)$$

$$(\forall x)R(x, k(x), f(x))$$

The latter is obviously a simpler formula.

Proposition

228

Suppose $\neg X'$ is a Skolemized version of the sentence $\neg X$. Then X is valid if and only if X' is valid.

Proof: X is valid iff $\{\neg X\}$ is not satisfiable iff $\{\neg X'\}$ is not satisfiable iff X' is valid.

Exercises:

1. Skolemize the following sentences
 - ▶ $(\forall x)(\exists y)(\forall z)(\exists w)R(x, y, z, w)$
 - ▶ $\neg(\exists x)[P(x) \supset (\forall x)P(x)]$
 - ▶ $\neg(\exists x)(\exists y)[P(x, y) \supset (\forall x)(\forall y)P(x, y)]$
 - ▶ $(\forall x)\{(\forall y)[((\forall z)P(x, y, z) \supset (\exists w)Q(x, y, w)) \supset R(x)] \supset S(x)\}$
2. implement Skolemization and integrate it into your prover

Proposition

228

Suppose $\neg X'$ is a Skolemized version of the sentence $\neg X$. Then X is valid if and only if X' is valid.

Proof: X is valid iff $\{\neg X\}$ is not satisfiable iff $\{\neg X'\}$ is not satisfiable iff X' is valid.

Exercises:

1. Skolemize the following sentences

- ▶ $(\forall x)(\exists y)(\forall z)(\exists w)R(x, y, z, w)$
- ▶ $\neg(\exists x)[P(x) \supset (\forall x)P(x)]$
- ▶ $\neg(\exists x)(\exists y)[P(x, y) \supset (\forall x)(\forall y)P(x, y)]$
- ▶ $(\forall x)\{(\forall y)[((\forall z)P(x, y, z) \supset (\exists w)Q(x, y, w)) \supset R(x)] \supset S(x)\}$

2. implement Skolemization and integrate it into your prover

Proposition

228

Suppose $\neg X'$ is a Skolemized version of the sentence $\neg X$. Then X is valid if and only if X' is valid.

Proof: X is valid iff $\{\neg X\}$ is not satisfiable iff $\{\neg X'\}$ is not satisfiable iff X' is valid.

Exercises:

1. Skolemize the following sentences
 - ▶ $(\forall x)(\exists y)(\forall z)(\exists w)R(x, y, z, w)$
 - ▶ $\neg(\exists x)[P(x) \supset (\forall x)P(x)]$
 - ▶ $\neg(\exists x)(\exists y)[P(x, y) \supset (\forall x)(\forall y)P(x, y)]$
 - ▶ $(\forall x)\{(\forall y)[((\forall z)P(x, y, z) \supset (\exists w)Q(x, y, w)) \supset R(x)] \supset S(x)\}$
2. implement Skolemization and integrate it into your prover

- ▶ Prenex form: Given a sentence X it is always possible to find an equivalent sentence X' in which all quantifiers come first.
- ▶ Sentences in prenex form that are also Solemized have a particularly simple structure.
- ▶ In order to simplify things with bound/free variables we assume that bound variables are initially renamed (named apart); this is easy to implement.

Definition — Named Apart

229

We say a formula has its variables *named apart* if no two quantifiers in Φ bind the same variable and no bound variable is also free.

- ▶ Prenex form: Given a sentence X it is always possible to find an equivalent sentence X' in which all quantifiers come first.
- ▶ Sentences in prenex form that are also Solemized have a particularly simple structure.
- ▶ In order to simplify things with bound/free variables we assume that bound variables are initially renamed (named apart); this is easy to implement.

Definition — Named Apart

229

We say a formula has its variables *named apart* if no two quantifiers in Φ bind the same variable and no bound variable is also free.

- We now present examples of some rewrite rules (equivalences) that can be used to rewrite any formula (that has been named apart) into an equivalent one in prenex form

Definition — Quantifier Rewrite Rules

230

$\neg(\exists x)A$	\equiv	$(\forall x)\neg A$	
$\neg(\forall x)A$	\equiv	$(\exists x)\neg A$	
$(\forall x)A \wedge B$	\equiv	$(\forall x)[A \wedge B]$	
$A \wedge (\forall x)B$	\equiv	$(\forall x)[A \wedge B]$	
$(\exists x)A \wedge B$	\equiv	$(\exists x)[A \wedge B]$	
$A \wedge (\exists x)B$	\equiv	$(\exists x)[A \wedge B]$	(for all connectives)
$(\forall x)A \supset B$	\equiv	$(\exists x)[A \supset B]$	
$A \supset (\forall x)B$	\equiv	$(\forall x)[A \supset B]$	
$(\exists x)A \supset B$	\equiv	$(\forall x)[A \supset B]$	
$A \supset (\exists x)B$	\equiv	$(\exists x)[A \supset B]$	
...	

- We now present examples of some rewrite rules (equivalences) that can be used to rewrite any formula (that has been named apart) into an equivalent one in prenex form

Definition — Quantifier Rewrite Rules

230

$\neg(\exists x)A$	\equiv	$(\forall x)\neg A$	
$\neg(\forall x)A$	\equiv	$(\exists x)\neg A$	
$(\forall x)A \wedge B$	\equiv	$(\forall x)[A \wedge B]$	
$A \wedge (\forall x)B$	\equiv	$(\forall x)[A \wedge B]$	
$(\exists x)A \wedge B$	\equiv	$(\exists x)[A \wedge B]$	
$A \wedge (\exists x)B$	\equiv	$(\exists x)[A \wedge B]$	(for all connectives)
$(\forall x)A \supset B$	\equiv	$(\exists x)[A \supset B]$	
$A \supset (\forall x)B$	\equiv	$(\forall x)[A \supset B]$	
$(\exists x)A \supset B$	\equiv	$(\forall x)[A \supset B]$	
$A \supset (\exists x)B$	\equiv	$(\exists x)[A \supset B]$	
...	

- ▶ We may first bring a formula to prenex form and then Skolemize or vice versa.
- ▶ In each case we end up with a formula in prenex form that only has universal quantifiers.

Proposition

231

There is an algorithm for converting a sentence Φ into a sentence Φ^ in prenex form, with only universal quantifiers, such that Φ is satisfiable if and only if Φ^* is satisfiable.*

More on clever pre-processing of formula sets:

Andreas Nonnengart and Christoph Weidenbach, Computing Small Clause Normal Forms, in Handbook of Automated Reasoning, 2001.

- ▶ We may first bring a formula to prenex form and then Skolemize or vice versa.
- ▶ In each case we end up with a formula in prenex form that only has universal quantifiers.

Proposition

231

There is an algorithm for converting a sentence Φ into a sentence Φ^ in prenex form, with only universal quantifiers, such that Φ is satisfiable if and only if Φ^* is satisfiable.*

More on clever pre-processing of formula sets:

Andreas Nonnengart and Christoph Weidenbach, Computing Small Clause Normal Forms, in Handbook of Automated Reasoning, 2001.

- ▶ We may first bring a formula to prenex form and then Skolemize or vice versa.
- ▶ In each case we end up with a formula in prenex form that only has universal quantifiers.

Proposition

231

There is an algorithm for converting a sentence Φ into a sentence Φ^ in prenex form, with only universal quantifiers, such that Φ is satisfiable if and only if Φ^* is satisfiable.*

More on clever pre-processing of formula sets:

Andreas Nonnengart and Christoph Weidenbach, Computing Small Clause Normal Forms, in Handbook of Automated Reasoning, 2001.

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:

• Eliminate equality and function symbols

• Eliminate existential quantifiers

• Eliminate universal quantifiers

• Eliminate negation

• Eliminate disjunction and conjunction

• Eliminate implication and equivalence

• Eliminate non-clausal normal forms

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.

2. Normalize:

- ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
- ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)(C_1, \dots, C_n)$ with the C_i being clauses.
- ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$((\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n)$$

- ▶ Drop universal quantifiers and treat them implicit:

$$(C_1, \dots, C_n)$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal, e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.

2. Normalize:

- ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
- ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
- ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$\langle (\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n \rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:
 - ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
 - ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
 - ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$\langle (\forall x_1) \dots (\forall x_n) C_1, \dots, (\forall x_1) \dots (\forall x_n) C_n \rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:
 - ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
 - ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
 - ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):
$$\langle (\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n \rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:
 - ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
 - ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
 - ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$\langle(\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n\rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:
 - ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
 - ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
 - ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$\langle(\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n\rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

What does this mean for resolution? We can strictly separate the reasoning process into parts:

1. Negate: replace the conjecture Φ to prove by $\neg\Phi$.
2. Normalize:
 - ▶ Rewrite $\neg\Phi$ into Skolemized prenex form $(\forall x_1) \dots (\forall x_n)\Psi$, where Ψ contains no quantifiers.
 - ▶ Convert the matrix into conjunctive normal form (clause form): $(\forall x_1) \dots (\forall x_n)\langle C_1, \dots, C_n \rangle$ with the C_i being clauses.
 - ▶ Convert this sentence into (by applying the equivalence $(\forall x)(A \wedge B) \equiv (\forall x)A \wedge (\forall x)B$):

$$\langle(\forall x_1) \dots (\forall x_n)C_1, \dots, (\forall x_1) \dots (\forall x_n)C_n\rangle$$

- ▶ Drop universal quantifiers and treat them implicit:

$$\langle C_1, \dots, C_n \rangle$$

This is a conjunction of clauses; the free-variables are implicitly assumed universal; e.g. they are always renamed.

3. Refute: Binary Resolution and Factoring

Exercises:

1. Give the remaining rewrite rules. Can you present them with the uniform notation approach?
2. Convert to prenex form:
 - ▶ $\{\neg(\exists x)(\forall y)P(x, y) \vee (\forall x)Q(x)\} \wedge \{(\forall x)A(x) \supset (\forall x)B(x)\}$
 - ▶ $\neg(\exists x)(\exists y)[P(x, y) \supset (\forall x)(\forall y)P(x, y)]$
 - ▶ $(\forall x)\{(\forall y)[((\forall z)P(x, y, z) \supset (\exists w)Q(x, y, w)) \supset R(x)] \supset S(x)\}$
3. Implement a program that realizes the normalization sketched above.
4. Adapt your resolution prover to the above approach.

Herbrand's Theorem (in his thesis, 1930)

- ▶ fundamental result: kind of constructive version of Gödel's completeness theorem
- ▶ theoretical and practical foundation for ATP
- ▶ involves notion of a *Herbrand expansion*: kind of finite approximation to a Herbrand model
- ▶ theorem says that X is a theorem iff some Herbrand expansion of X is a propositional tautology
- ▶ thus, the theorem splits the prove process into
 - ▶ the enumeration resp. systematic generation of Herbrand expansions, and
 - ▶ the checking whether a Herbrand expansion is propositional tautology (which is decidable)
- ▶ Herbrand's constructive perspective on the field is in contrast with the semantic methods applied in this lecture
- ▶ has led to decidability results for several subclasses of FOL
- ▶ possible to prove Herbrand's theorem by Abstract Consistency

For more information on Herbrand's work:

- ▶ C.P. Wirth, J. Siekmann, C. Benzmüller, and S. Autexier,
Jacques Herbrand: Life, Logic, and Automated Deduction. In
D. Gabbay, J. Woods (eds.), The Handbook of the History of
Logic, Volume 5 — Logic and Sets from Russell to Church:
Logic. 2009.

<http://linkinghub.elsevier.com/retrieve/pii/S1874585709700093>

- ▶ Pre-print:
C.-P. Wirth, J. Siekmann, C. Benzmüller, S. Autexier,
Lectures on Jacques Herbrand as a Logician. Seki Report
SR-2009-01 (ISSN 1437-4447), Saarland University, 2009. (ii
+ 70 pages)

<http://wwwags.uni-sb.de/~chris/papers/R43.pdf>

- ▶ Alternative version available in arXiv:

<http://arxiv.org/abs/0902.4682>

Definition — Validity Functional

232

The sentence X' is a *validity functional form* of X if $\neg X'$ is a Skolemized version of $\neg X$.

Remarks: Terminology becomes clear if one considers: X is valid iff X' is. Also note that Skolemization of $\neg X$ removes all its essentially existential variables, i.e. the essentially universal quantifiers of X . The validity functional form of a given sentence contains thus only essentially existential quantifiers. The idea now is to appropriately 'instantiate/replace' these (essentially) existentially quantified formulas.

Definition — Validity Functional

232

The sentence X' is a *validity functional form* of X if $\neg X'$ is a Skolemized version of $\neg X$.

Remarks: Terminology becomes clear if one considers: X is valid iff X' is. Also note that Skolemization of $\neg X$ removes all its essentially existential variables, i.e. the essentially universal quantifiers of X . The validity functional form of a given sentence contains thus only essentially existential quantifiers. The idea now is to appropriately 'instantiate/replace' these (essentially) existentially quantified formulas.

Definition — Validity Functional

232

The sentence X' is a *validity functional form* of X if $\neg X'$ is a Skolemized version of $\neg X$.

Remarks: Terminology becomes clear if one considers: X is valid iff X' is. Also note that Skolemization of $\neg X$ removes all its essentially existential variables, i.e. the essentially universal quantifiers of X . The validity functional form of a given sentence contains thus only essentially existential quantifiers. The idea now is to appropriately 'instantiate/replace' these (essentially) existentially quantified formulas.

Definition — Validity Functional

232

The sentence X' is a *validity functional form* of X if $\neg X'$ is a Skolemized version of $\neg X$.

Remarks: Terminology becomes clear if one considers: X is valid iff X' is. Also note that Skolemization of $\neg X$ removes all its essentially existential variables, i.e. the essentially universal quantifiers of X . The validity functional form of a given sentence contains thus only essentially existential quantifiers. The idea now is to appropriately 'instantiate/replace' these (essentially) existentially quantified formulas.

Definition — Validity Functional

232

The sentence X' is a *validity functional form* of X if $\neg X'$ is a Skolemized version of $\neg X$.

Remarks: Terminology becomes clear if one considers: X is valid iff X' is. Also note that Skolemization of $\neg X$ removes all its essentially existential variables, i.e. the essentially universal quantifiers of X . The validity functional form of a given sentence contains thus only essentially existential quantifiers. The idea now is to appropriately 'instantiate/replace' these (essentially) existentially quantified formulas.

Definition — Herbrand Universe

233

Let X be a sentence. The *Herbrand universe* for X is the set of all closed terms constructed from the constant and function symbols of X . (If X contains no constant symbols, add an arbitrary new constant symbol c_0).

Example

234

The Herbrand universe of $(\forall x)[(\exists y)R(x, y) \supset R(b, f(x))]$ is

$$\{b, f(b), f(f(b)), \dots\}$$

The Herbrand universe of $(\forall x)(\exists y)R(x, y)$ is $\{c_0\}$.

Definition — Herbrand Domain

235

A *Herbrand domain* for a sentence X is a finite, non-empty subset of the Herbrand universe of X .

Definition — Herbrand Universe

233

Let X be a sentence. The *Herbrand universe* for X is the set of all closed terms constructed from the constant and function symbols of X . (If X contains no constant symbols, add an arbitrary new constant symbol c_0).

Example

234

The Herbrand universe of $(\forall x)[(\exists y)R(x, y) \supset R(b, f(x))]$ is

$$\{b, f(b), f(f(b)), \dots\}$$

The Herbrand universe of $(\forall x)(\exists y)R(x, y)$ is $\{c_0\}$.

Definition — Herbrand Domain

235

A *Herbrand domain* for a sentence X is a finite, non-empty subset of the Herbrand universe of X .

Definition — Herbrand Universe

233

Let X be a sentence. The *Herbrand universe* for X is the set of all closed terms constructed from the constant and function symbols of X . (If X contains no constant symbols, add an arbitrary new constant symbol c_0).

Example

234

The Herbrand universe of $(\forall x)[(\exists y)R(x, y) \supset R(b, f(x))]$ is

$$\{b, f(b), f(f(b)), \dots\}$$

The Herbrand universe of $(\forall x)(\exists y)R(x, y)$ is $\{c_0\}$.

Definition — Herbrand Domain

235

A *Herbrand domain* for a sentence X is a finite, non-empty subset of the Herbrand universe of X .

Main idea of a *Herbrand Expansion*:

- ▶ (essentially) existentially quantified formulas become disjunctions of instances over the Herbrand universe, and
- ▶ (essentially) universally quantified formulas become conjunctions of instances over the Herbrand universe.

For Herbrand's theorem we only need to address (essentially) existentially quantified formulas; we nevertheless also give the γ -case below.

Main idea of a *Herbrand Expansion*:

- ▶ (essentially) existentially quantified formulas become disjunctions of instances over the Herbrand universe, and
- ▶ (essentially) universally quantified formulas become conjunctions of instances over the Herbrand universe.

For Herbrand's theorem we only need to address (essentially) existentially quantified formulas; we nevertheless also give the γ -case below.

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Definition — Herbrand Expansion

236

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon(X, D)$, is defined as follows:

1. If X is a literal, then $\epsilon(X, D) = X$.
2. If X is of form $\neg\neg X'$, then $\epsilon(X, D) = X'$.
3. If $X = \alpha$, then $\epsilon(X, D) = \epsilon(\alpha_1, D) \wedge \epsilon(\alpha_2, D)$.
4. If $X = \beta$, then $\epsilon(X, D) = \epsilon(\beta_1, D) \vee \epsilon(\beta_2, D)$.
5. If $X = \gamma$ then, $\epsilon(X, D) = \epsilon(\gamma(t_1), D) \wedge \dots \wedge \epsilon(\gamma(t_n), D)$.
6. If $X = \delta$ then, $\epsilon(X, D) = \epsilon(\delta(t_1), D) \vee \dots \vee \epsilon(\delta(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Theorem — Herbrand's Theorem

237

A sentence Y is valid if and only if some Herbrand expansion of Y is a tautology.

Proof: . . . using Abstract Consistency, not here . . .

Example

238

$$Y := (\forall z)(\exists w)(\forall x)[(\forall y)R(x, y) \supset R(w, z)]$$

Validity functional form: $X := (\exists w)[(\forall y)R(f(w), y) \supset R(w, c)]$

Herbrand universe for X : $\{c, f(c), f(f(c)), \dots\}$

A Herbrand domain for X : $\{c\}$

Herbrand expansion of X for $\{c\}$: $\neg R(f(c), c) \vee R(c, c)$

A Herbrand domain for X : $\{c, f(c)\}$

Herbrand expansion (now a tautology) of X for $\{c, f(c)\}$:

$$\begin{aligned} & \neg R(f(c), c) \vee \neg R(f(c), f(c)) \vee R(c, c) \vee \\ & \neg R(f(f(c)), c) \vee \neg R(f(f(c)), f(c)) \vee R(f(c), c) \end{aligned}$$

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

Definition — Herbrand Expansion (conventional def.) 239

Let $D = \{t_1, \dots, t_n\}$ be a non-empty set of closed terms and let X be a sentence. The *Herbrand Expansion of X over (the finite domain) D* , denoted $\epsilon'(X, D)$, is defined as follows:

1. If X is atomic, then $\epsilon'(X, D) = X$.
2. $\epsilon'(\neg Z, D) = \neg \epsilon'(Z, D)$.
3. $\epsilon'(Z \circ W, D) = \epsilon'(Z, D) \circ \epsilon'(W, D)$.
4. $\epsilon'((\forall x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \wedge \dots \wedge \epsilon'(\Phi(t_n), D)$.
5. $\epsilon'((\exists x)\Phi(x), D) = \epsilon'(\Phi(t_1), D) \vee \dots \vee \epsilon'(\Phi(t_n), D)$.

A *Herbrand expansion* of a sentence Y is a Herbrand expansion of X over D , where X is a validity functional form of Y and where D is any Herbrand domain for X .

Exercises: Make up some formulas and present their Herbrand expansions.

See papers by Martin Davis (including pointers to literature), e.g.

<http://cs.nyu.edu/cs/faculty/davism/early.ps>

Propositional Logic

- 1929** Presburger showed that the first-order theory of addition in arithmetic of integers is decidable.
- 1954** Martin Davis programmed this algorithm for the vacuum tube computer at the Institute for Advanced Study in Princeton.
- 1957** Newell, Shaw and Simon report on experiments with their 'Logic Theory Machine' on automating the propositional calculus of Principia Mathematica. In 1987 Wang and Gao presented a complete and more efficient Gentzen-style system for the propositional calculus of Principia Mathematica.
- 1959** Gelernter presents his Geometry Machine, which was pretty much in the spirit of Newell, Shaw and Simon's work.

First-Order Logic

- 1920-30** Foundational work by Skolem, Herbrand, Gentzen, etc.
- 1957** Talk by Abraham Robinson at Cornell University in which he pointed to the usefulness of Skolem functions and Herbrand's theorem for general purpose theorem provers.
- 1960** Gilmore presents a program that is capable of proving some simple theorems.
- 1960** Prawitz, Prawitz and Voghera implement a prover based on semantic tableaux.
- 1960,1962** Martin Davis and Hilary Putnam realized the crucial role of efficient propositional reasoning also for first-order logic; their method instantiated first-order logic formulas to propositional logic formulas (enumeration of the Herbrand universe) and then used satisfiability checking at the propositional level; their method was later improved in collaboration with Logemann and Loveland — you may have heard of the DPLL method (see talk of J. Otten).

1960s Prawitz proposed methods that avoided the enumeration of the Herbrand universe; these ideas were again taken up by Davis and colleagues

1965 J.A.Robinson announced the resolution method (including unification) in his landmark paper; after this became a mature field.

Influential, more recent work:

- ▶ Superposition (Bachmair, Ganzinger, Rewrite-based equational theorem proving with selection and simplification, J.Logic and Computation 4 (1994): 217-247)

To get really familiar with the area: Handbook of Automated Reasoning, J.A. Robinson and A. Voronkov, eds., Elsevier, 2001.