Miscellaneous FOL Examples

October 24, 2026

Contents

1 Natural numbers **1**

2 Examples for the manual "Introduction to Isabelle" **2**
2.0.1 Some simple backward proofs 2
2.0.2 Demonstration of *fast* 2
2.0.3 Derivation of conjunction elimination rule 2
2.1 Derived rules involving definitions 3

3 Theory of the natural numbers: Peano's axioms, primitive recursion **3**
3.1 Proofs about the natural numbers 4

4 Theory of the natural numbers: Peano's axioms, primitive recursion **4**

5 Intuitionistic FOL: Examples from The Foundation of a Generic Theorem Prover **5**
5.1 Examples with quantifiers 6

6 First-Order Logic: PROLOG examples **7**

7 Intuitionistic First-Order Logic **8**
7.1 Lemmas for the propositional double-negation translation . . 9
7.2 de Bruijn formulae 9
7.3 Intuitionistic FOL: propositional problems based on Pelletier. 10
7.4 11. Proved in each direction (incorrectly, says Pelletier!!) . . . 11

8 Examples with quantifiers **12**
8.1 The converse is classical in the following implications 12
8.2 The following are not constructively valid! 12
8.3 Hard examples with quantifiers 13

9 First-Order Logic: propositional examples (intuitionistic version) 17

10 First-Order Logic: quantifier examples (intuitionistic version) 19

11 Classical Predicate Calculus Problems 20
11.0.1 If and only if 20
11.1 Pelletier's examples 21
11.2 Classical Logic: examples with quantifiers 22
11.3 Problems requiring quantifier duplication 23
11.4 Hard examples with quantifiers 23
11.5 Problems (mainly) involving equality or functions 27

12 First-Order Logic: propositional examples (classical version) 31

13 First-Order Logic: quantifier examples (classical version) 33
13.1 Negation Normal Form 35
13.1.1 de Morgan laws 35
13.1.2 Pushing in the existential quantifiers 35
13.1.3 Pushing in the universal quantifiers 35

14 First-Order Logic: the 'if' example 36

1 Natural numbers

theory *Natural-Numbers*
imports *FOL*
begin

Theory of the natural numbers: Peano's axioms, primitive recursion. (Modernized version of Larry Paulson's theory "Nat".)

typedecl *nat*
instance *nat* :: ‹*term*› ⟨*proof*⟩

axiomatization
Zero :: ‹*nat*› (‹*0*›) **and**
Suc :: ‹*nat* => *nat*› **and**
rec :: ‹[*nat*, *'a*, [*nat*, *'a*] => *'a*] => *'a*›
where
induct [*case-names 0 Suc*, *induct type*: *nat*]:
‹*P*(*0*) ==> (!!*x*. *P*(*x*) ==> *P*(*Suc*(*x*))) ==> *P*(*n*)› **and**
Suc-inject: ‹*Suc*(*m*) = *Suc*(*n*) ==> *m* = *n*› **and**
Suc-neq-0: ‹*Suc*(*m*) = *0* ==> *R*› **and**
rec-0: ‹*rec*(*0*, *a*, *f*) = *a*› **and**

rec-Suc: ‹$rec(Suc(m),\ a,\ f) = f(m,\ rec(m,\ a,\ f))$›

lemma *Suc-n-not-n*: ‹$Suc(k) \neq k$›
⟨*proof*⟩

definition *add* :: ‹$nat \Rightarrow nat \Rightarrow nat$› (**infixl** ‹+› *60*)
where ‹$m + n = rec(m,\ n,\ \lambda x\ y.\ Suc(y))$›

lemma *add-0* [*simp*]: ‹$0 + n = n$›
⟨*proof*⟩

lemma *add-Suc* [*simp*]: ‹$Suc(m) + n = Suc(m + n)$›
⟨*proof*⟩

lemma *add-assoc*: ‹$(k + m) + n = k + (m + n)$›
⟨*proof*⟩

lemma *add-0-right*: ‹$m + 0 = m$›
⟨*proof*⟩

lemma *add-Suc-right*: ‹$m + Suc(n) = Suc(m + n)$›
⟨*proof*⟩

lemma
assumes ‹$!!n.\ f(Suc(n)) = Suc(f(n))$›
shows ‹$f(i + j) = i + f(j)$›
⟨*proof*⟩

end

2 Examples for the manual “Introduction to Isabelle”

theory *Intro*
imports *FOL*
begin

2.0.1 Some simple backward proofs

lemma *mythm*: ‹$P \vee P \longrightarrow P$›
⟨*proof*⟩

lemma ‹$(P \wedge Q) \vee R \longrightarrow (P \vee R)$›
⟨*proof*⟩

Correct version, delaying use of *spec* until last.

lemma ‹$(\forall x\ y.\ P(x,y)) \longrightarrow (\forall z\ w.\ P(w,z))$›

⟨*proof*⟩

2.0.2 Demonstration of *fast*

lemma ‹$(\exists\, y.\ \forall\, x.\ J(y,x) \longleftrightarrow \neg\ J(x,x)) \longrightarrow \neg\ (\forall\, x.\ \exists\, y.\ \forall\, z.\ J(z,y) \longleftrightarrow \neg\ J(z,x))$›
⟨*proof*⟩

lemma ‹$\forall\, x.\ P(x,f(x)) \longleftrightarrow (\exists\, y.\ (\forall\, z.\ P(z,y) \longrightarrow P(z,f(x))) \wedge P(x,y))$›
⟨*proof*⟩

2.0.3 Derivation of conjunction elimination rule

lemma
  assumes *major*: ‹$P \wedge Q$›
    and *minor*: ‹$[\![P;\ Q]\!] \Longrightarrow R$›
  shows ‹R›
⟨*proof*⟩

2.1 Derived rules involving definitions

Derivation of negation introduction

lemma
  assumes ‹$P \Longrightarrow \mathit{False}$›
  shows ‹$\neg\ P$›
⟨*proof*⟩

lemma
  assumes *major*: ‹$\neg\ P$›
    and *minor*: ‹P›
  shows ‹R›
⟨*proof*⟩

Alternative proof of the result above

lemma
  assumes *major*: ‹$\neg\ P$›
    and *minor*: ‹P›
  shows ‹R›
⟨*proof*⟩

end

3 Theory of the natural numbers: Peano's axioms, primitive recursion

theory *Nat*
  imports *FOL*
begin

typedecl *nat*
instance *nat* :: ‹*term*› ⟨*proof*⟩

axiomatization
  Zero :: ‹*nat*› (‹*0*›) **and**
  Suc :: ‹*nat* ⇒ *nat*› **and**
  rec :: ‹[*nat*, *'a*, [*nat*, *'a*] ⇒ *'a*] ⇒ *'a*›
where
  induct: ‹⟦*P*(*0*); ⋀*x*. *P*(*x*) ⟹ *P*(*Suc*(*x*))⟧ ⟹ *P*(*n*)› **and**
  Suc-inject: ‹*Suc*(*m*)=*Suc*(*n*) ⟹ *m*=*n*› **and**
  Suc-neq-0: ‹*Suc*(*m*)=*0* ⟹ *R*› **and**
  rec-0: ‹*rec*(*0*,*a*,*f*) = *a*› **and**
  rec-Suc: ‹*rec*(*Suc*(*m*), *a*, *f*) = *f*(*m*, *rec*(*m*,*a*,*f*))›

definition *add* :: ‹[*nat*, *nat*] ⇒ *nat*› (**infixl** ‹+› *60*)
  where ‹*m* + *n* ≡ *rec*(*m*, *n*, λ*x* *y*. *Suc*(*y*))›

3.1 Proofs about the natural numbers

lemma *Suc-n-not-n*: ‹*Suc*(*k*) ≠ *k*›
⟨*proof*⟩

lemma ‹(*k*+*m*)+*n* = *k*+(*m*+*n*)›
⟨*proof*⟩

lemma *add-0* [*simp*]: ‹*0*+*n* = *n*›
⟨*proof*⟩

lemma *add-Suc* [*simp*]: ‹*Suc*(*m*)+*n* = *Suc*(*m*+*n*)›
⟨*proof*⟩

lemma *add-assoc*: ‹(*k*+*m*)+*n* = *k*+(*m*+*n*)›
⟨*proof*⟩

lemma *add-0-right*: ‹*m*+*0* = *m*›
⟨*proof*⟩

lemma *add-Suc-right*: ‹*m*+*Suc*(*n*) = *Suc*(*m*+*n*)›
⟨*proof*⟩

lemma
  assumes *prem*: ‹⋀*n*. *f*(*Suc*(*n*)) = *Suc*(*f*(*n*))›
  shows ‹*f*(*i*+*j*) = *i*+*f*(*j*)›
⟨*proof*⟩

end

4 Theory of the natural numbers: Peano's axioms, primitive recursion

theory *Nat-Class*
imports *FOL*
begin

This is an abstract version of `Nat.thy`. Instead of axiomatizing a single type *nat*, it defines the class of all these types (up to isomorphism).

Note: The *rec* operator has been made *monomorphic*, because class axioms cannot contain more than one type variable.

class *nat* =
fixes *Zero* :: ‹'*a*› (‹*0*›)
and *Suc* :: ‹'*a* ⇒ '*a*›
and *rec* :: ‹'*a* ⇒ '*a* ⇒ ('*a* ⇒ '*a* ⇒ '*a*) ⇒ '*a*›
assumes *induct*: ‹*P*(*0*) ⟹ (⋀*x*. *P*(*x*) ⟹ *P*(*Suc*(*x*))) ⟹ *P*(*n*)›
and *Suc-inject*: ‹*Suc*(*m*) = *Suc*(*n*) ⟹ *m* = *n*›
and *Suc-neq-Zero*: ‹*Suc*(*m*) = *0* ⟹ *R*›
and *rec-Zero*: ‹*rec*(*0*, *a*, *f*) = *a*›
and *rec-Suc*: ‹*rec*(*Suc*(*m*), *a*, *f*) = *f*(*m*, *rec*(*m*, *a*, *f*))›
begin

definition *add* :: ‹'*a* ⇒ '*a* ⇒ '*a*› (**infixl** ‹+› *60*)
where ‹*m* + *n* = *rec*(*m*, *n*, λ*x* *y*. *Suc*(*y*))›

lemma *Suc-n-not-n*: ‹*Suc*(*k*) ≠ (*k*::'*a*)›
⟨*proof*⟩

lemma ‹(*k* + *m*) + *n* = *k* + (*m* + *n*)›
⟨*proof*⟩

lemma *add-Zero* [*simp*]: ‹*0* + *n* = *n*›
⟨*proof*⟩

lemma *add-Suc* [*simp*]: ‹*Suc*(*m*) + *n* = *Suc*(*m* + *n*)›
⟨*proof*⟩

lemma *add-assoc*: ‹(*k* + *m*) + *n* = *k* + (*m* + *n*)›
⟨*proof*⟩

lemma *add-Zero-right*: ‹*m* + *0* = *m*›
⟨*proof*⟩

lemma *add-Suc-right*: ‹*m* + *Suc*(*n*) = *Suc*(*m* + *n*)›
⟨*proof*⟩

lemma
assumes *prem*: ‹⋀*n*. *f*(*Suc*(*n*)) = *Suc*(*f*(*n*))›

shows ‹$f(i + j) = i + f(j)$›
⟨*proof*⟩

end

end

5 Intuitionistic FOL: Examples from The Foundation of a Generic Theorem Prover

theory *Foundation*
imports *IFOL*
begin

lemma ‹$A \wedge B \longrightarrow (C \longrightarrow A \wedge C)$›
⟨*proof*⟩

A form of conj-elimination

lemma
assumes ‹$A \wedge B$›
and ‹$A \Longrightarrow B \Longrightarrow C$›
shows ‹C›
⟨*proof*⟩

lemma
assumes ‹$\bigwedge A.\ \neg\ \neg\ A \Longrightarrow A$›
shows ‹$B \vee \neg\ B$›
⟨*proof*⟩

lemma
assumes ‹$\bigwedge A.\ \neg\ \neg\ A \Longrightarrow A$›
shows ‹$B \vee \neg\ B$›
⟨*proof*⟩

lemma
assumes ‹$A \vee \neg\ A$›
and ‹$\neg\ \neg\ A$›
shows ‹A›
⟨*proof*⟩

5.1 Examples with quantifiers

lemma
assumes ‹$\forall\, z.\ G(z)$›
shows ‹$\forall\, z.\ G(z) \vee H(z)$›
⟨*proof*⟩

lemma ‹$\forall x.\ \exists y.\ x = y$›
⟨*proof*⟩

lemma ‹$\exists y.\ \forall x.\ x = y$›
⟨*proof*⟩

Parallel lifting example.

lemma ‹$\exists u.\ \forall x.\ \exists v.\ \forall y.\ \exists w.\ P(u,x,v,y,w)$›
⟨*proof*⟩

lemma
assumes ‹$(\exists z.\ F(z)) \wedge B$›
shows ‹$\exists z.\ F(z) \wedge B$›
⟨*proof*⟩

A bigger demonstration of quantifiers – not in the paper.

lemma ‹$(\exists y.\ \forall x.\ Q(x,y)) \longrightarrow (\forall x.\ \exists y.\ Q(x,y))$›
⟨*proof*⟩

end

6 First-Order Logic: PROLOG examples

theory *Prolog*
imports *FOL*
begin

typedecl *'a list*
instance *list* :: (‹*term*›) ‹*term*› ⟨*proof*⟩

axiomatization
Nil :: ‹*'a list*› **and**
Cons :: ‹[*'a*, *'a list*]=> *'a list*› (**infixr** ‹:› *60*) **and**
app :: ‹[*'a list*, *'a list*, *'a list*] => *o*› **and**
rev :: ‹[*'a list*, *'a list*] => *o*›
where
appNil: ‹*app*(*Nil*,*ys*,*ys*)› **and**
appCons: ‹*app*(*xs*,*ys*,*zs*) ==> *app*(*x*:*xs*, *ys*, *x*:*zs*)› **and**
revNil: ‹*rev*(*Nil*,*Nil*)› **and**
revCons: ‹[| *rev*(*xs*,*ys*); *app*(*ys*, *x*:*Nil*, *zs*) |] ==> *rev*(*x*:*xs*, *zs*)›

schematic-goal ‹*app*(*a*:*b*:*c*:*Nil*, *d*:*e*:*Nil*, *?x*)›
⟨*proof*⟩

schematic-goal ‹*app*(*?x*, *c*:*d*:*Nil*, *a*:*b*:*c*:*d*:*Nil*)›
⟨*proof*⟩

schematic-goal ‹*app*(*?x*, *?y*, *a*:*b*:*c*:*d*:*Nil*)›

⟨*proof*⟩

lemmas *rules* = *appNil appCons revNil revCons*

schematic-goal ‹*rev*(*a*:*b*:*c*:*d*:*Nil*, *?x*)›
⟨*proof*⟩

schematic-goal ‹*rev*(*a*:*b*:*c*:*d*:*e*:*f*:*g*:*h*:*i*:*j*:*k*:*l*:*m*:*n*:*Nil*, *?w*)›
⟨*proof*⟩

schematic-goal ‹*rev*(*?x*, *a*:*b*:*c*:*Nil*)›
⟨*proof*⟩

⟨*ML*⟩

schematic-goal ‹*rev*(*?x*, *a*:*b*:*c*:*Nil*)›
⟨*proof*⟩

schematic-goal ‹*rev*(*a*:*?x*:*c*:*?y*:*Nil*, *d*:*?z*:*b*:*?u*)›
⟨*proof*⟩

schematic-goal ‹*rev*(*a*:*b*:*c*:*d*:*e*:*f*:*g*:*h*:*i*:*j*:*k*:*l*:*m*:*n*:*o*:*p*:*Nil*, *?w*)›
⟨*proof*⟩

schematic-goal ‹*a*:*b*:*c*:*d*:*e*:*f*:*g*:*h*:*i*:*j*:*k*:*l*:*m*:*n*:*o*:*p*:*Nil* = *?x* ∧ *app*(*?x*, *?x*, *?y*) ∧ *rev*(*?y*, *?w*)›
⟨*proof*⟩

end

7 Intuitionistic First-Order Logic

theory *Intuitionistic*
imports *IFOL*
begin

Metatheorem (for *propositional* formulae): P is classically provable iff $\neg\neg P$ is intuitionistically provable. Therefore $\neg P$ is classically provable iff it is intuitionistically provable.

Proof: Let Q be the conjunction of the propositions $A \vee \neg A$, one for each atom A in P. Now $\neg\neg Q$ is intuitionistically provable because $\neg\neg(A \vee \neg A)$ is and because double-negation distributes over conjunction. If P is provable classically, then clearly $Q \to P$ is provable intuitionistically, so $\neg\neg(Q \to P)$

is also provable intuitionistically. The latter is intuitionistically equivalent to $\neg\neg Q \to \neg\neg P$, hence to $\neg\neg P$, since $\neg\neg Q$ is intuitionistically provable. Finally, if P is a negation then $\neg\neg P$ is intuitionstically equivalent to P. [Andy Pitts]

lemma ‹$\neg\ \neg\ (P \wedge Q) \longleftrightarrow \neg\ \neg\ P \wedge \neg\ \neg\ Q$›
⟨*proof*⟩

lemma ‹$\neg\ \neg\ ((\neg\ P \longrightarrow Q) \longrightarrow (\neg\ P \longrightarrow \neg\ Q) \longrightarrow P)$›
⟨*proof*⟩

Double-negation does NOT distribute over disjunction.

lemma ‹$\neg\ \neg\ (P \longrightarrow Q) \longleftrightarrow (\neg\ \neg\ P \longrightarrow \neg\ \neg\ Q)$›
⟨*proof*⟩

lemma ‹$\neg\ \neg\ \neg\ P \longleftrightarrow \neg\ P$›
⟨*proof*⟩

lemma ‹$\neg\ \neg\ ((P \longrightarrow Q \vee R) \longrightarrow (P \longrightarrow Q) \vee (P \longrightarrow R))$›
⟨*proof*⟩

lemma ‹$(P \longleftrightarrow Q) \longleftrightarrow (Q \longleftrightarrow P)$›
⟨*proof*⟩

lemma ‹$((P \longrightarrow (Q \vee (Q \longrightarrow R))) \longrightarrow R) \longrightarrow R$›
⟨*proof*⟩

lemma
‹$(((G \longrightarrow A) \longrightarrow J) \longrightarrow D \longrightarrow E) \longrightarrow (((H \longrightarrow B) \longrightarrow I) \longrightarrow C \longrightarrow J)$
$\longrightarrow (A \longrightarrow H) \longrightarrow F \longrightarrow G \longrightarrow (((C \longrightarrow B) \longrightarrow I) \longrightarrow D) \longrightarrow (A \longrightarrow C)$
$\longrightarrow (((F \longrightarrow A) \longrightarrow B) \longrightarrow I) \longrightarrow E$›
⟨*proof*⟩

Admissibility of the excluded middle for negated formulae

lemma ‹$(P \vee \neg P \longrightarrow \neg Q) \longrightarrow \neg Q$›
⟨*proof*⟩

The same in a more general form, no ex falso quodlibet

lemma ‹$(P \vee (P \longrightarrow R) \longrightarrow Q \longrightarrow R) \longrightarrow Q \longrightarrow R$›
⟨*proof*⟩

7.1 Lemmas for the propositional double-negation translation

lemma ‹$P \longrightarrow \neg\ \neg\ P$›
⟨*proof*⟩

lemma ‹$\neg\ \neg\ (\neg\ \neg\ P \longrightarrow P)$›
⟨*proof*⟩

lemma ‹¬ ¬ P ∧ ¬ ¬ (P ⟶ Q) ⟶ ¬ ¬ Q›
⟨*proof*⟩

The following are classically but not constructively valid. The attempt to prove them terminates quickly!

lemma ‹((P ⟶ Q) ⟶ P) ⟶ P›
⟨*proof*⟩

lemma ‹(P ∧ Q ⟶ R) ⟶ (P ⟶ R) ∨ (Q ⟶ R)›
⟨*proof*⟩

7.2 de Bruijn formulae

de Bruijn formula with three predicates

lemma
‹((P ⟷ Q) ⟶ P ∧ Q ∧ R) ∧
(Q ⟷ R) ⟶ P ∧ Q ∧ R) ∧
(R ⟷ P) ⟶ P ∧ Q ∧ R) ⟶ P ∧ Q ∧ R›
⟨*proof*⟩

de Bruijn formula with five predicates

lemma
‹((P ⟷ Q) ⟶ P ∧ Q ∧ R ∧ S ∧ T) ∧
(Q ⟷ R) ⟶ P ∧ Q ∧ R ∧ S ∧ T) ∧
(R ⟷ S) ⟶ P ∧ Q ∧ R ∧ S ∧ T) ∧
(S ⟷ T) ⟶ P ∧ Q ∧ R ∧ S ∧ T) ∧
(T ⟷ P) ⟶ P ∧ Q ∧ R ∧ S ∧ T) ⟶ P ∧ Q ∧ R ∧ S ∧ T›
⟨*proof*⟩

Problems from of Sahlin, Franzen and Haridi, An Intuitionistic Predicate Logic Theorem Prover. J. Logic and Comp. 2 (5), October 1992, 619-656.

Problem 1.1

lemma
‹(∀ x. ∃ y. ∀ z. $p(x)$ ∧ $q(y)$ ∧ $r(z)$) ⟷
(∀ z. ∃ y. ∀ x. $p(x)$ ∧ $q(y)$ ∧ $r(z)$)›
⟨*proof*⟩

Problem 3.1

lemma ‹¬ (∃ x. ∀ y. $mem(y,x)$ ⟷ ¬ $mem(x,x)$)›
⟨*proof*⟩

Problem 4.1: hopeless!

lemma
‹(∀ x. $p(x)$ ⟶ $p(h(x))$ ∨ $p(g(x))$) ∧ (∃ x. $p(x)$) ∧ (∀ x. ¬ $p(h(x))$)
⟶ (∃ x. $p(g(g(g(g(g(g(g(x)))))))$›
⟨*proof*⟩

7.3 Intuitionistic FOL: propositional problems based on Pelletier.

¬¬1

lemma ‹¬ ¬ ((P ⟶ Q) ⟷ (¬ Q ⟶ ¬ P))›
⟨*proof*⟩

¬¬2

lemma ‹¬ ¬ (¬ ¬ P ⟷ P)›
⟨*proof*⟩

3

lemma ‹¬ (P ⟶ Q) ⟶ (Q ⟶ P)›
⟨*proof*⟩

¬¬4

lemma ‹¬ ¬ ((¬ P ⟶ Q) ⟷ (¬ Q ⟶ P))›
⟨*proof*⟩

¬¬5

lemma ‹¬ ¬ ((P ∨ Q ⟶ P ∨ R) ⟶ P ∨ (Q ⟶ R))›
⟨*proof*⟩

¬¬6

lemma ‹¬ ¬ (P ∨ ¬ P)›
⟨*proof*⟩

¬¬7

lemma ‹¬ ¬ (P ∨ ¬ ¬ ¬ P)›
⟨*proof*⟩

¬¬8. Peirce's law

lemma ‹¬ ¬ (((P ⟶ Q) ⟶ P) ⟶ P)›
⟨*proof*⟩

9

lemma ‹((P ∨ Q) ∧ (¬ P ∨ Q) ∧ (P ∨ ¬ Q)) ⟶ ¬ (¬ P ∨ ¬ Q)›
⟨*proof*⟩

10

lemma ‹(Q ⟶ R) ⟶ (R ⟶ P ∧ Q) ⟶ (P ⟶ (Q ∨ R)) ⟶ (P ⟷ Q)›
⟨*proof*⟩

7.4 11. Proved in each direction (incorrectly, says Pelletier!!)

lemma ‹$P \longleftrightarrow P$›
⟨*proof*⟩

¬¬12. Dijkstra's law

lemma ‹$\neg\ \neg\ (((P \longleftrightarrow Q) \longleftrightarrow R) \longleftrightarrow (P \longleftrightarrow (Q \longleftrightarrow R)))$›
⟨*proof*⟩

lemma ‹$((P \longleftrightarrow Q) \longleftrightarrow R) \longrightarrow \neg\ \neg\ (P \longleftrightarrow (Q \longleftrightarrow R))$›
⟨*proof*⟩

13. Distributive law

lemma ‹$P \vee (Q \wedge R) \longleftrightarrow (P \vee Q) \wedge (P \vee R)$›
⟨*proof*⟩

¬¬14

lemma ‹$\neg\ \neg\ ((P \longleftrightarrow Q) \longleftrightarrow ((Q \vee \neg\ P) \wedge (\neg\ Q \vee P)))$›
⟨*proof*⟩

¬¬15

lemma ‹$\neg\ \neg\ ((P \longrightarrow Q) \longleftrightarrow (\neg\ P \vee Q))$›
⟨*proof*⟩

¬¬16

lemma ‹$\neg\ \neg\ ((P \longrightarrow Q) \vee (Q \longrightarrow P))$›
⟨*proof*⟩

¬¬17

lemma ‹$\neg\ \neg\ (((P \wedge (Q \longrightarrow R)) \longrightarrow S) \longleftrightarrow ((\neg\ P \vee Q \vee S) \wedge (\neg\ P \vee \neg\ R \vee S)))$›
⟨*proof*⟩

Dijkstra's "Golden Rule"

lemma ‹$(P \wedge Q) \longleftrightarrow P \longleftrightarrow Q \longleftrightarrow (P \vee Q)$›
⟨*proof*⟩

8 Examples with quantifiers

8.1 The converse is classical in the following implications ...

lemma ‹$(\exists x.\ P(x) \longrightarrow Q) \longrightarrow (\forall x.\ P(x)) \longrightarrow Q$›
⟨*proof*⟩

lemma ‹$((\forall x.\ P(x)) \longrightarrow Q) \longrightarrow \neg\ (\forall x.\ P(x) \wedge \neg\ Q)$›
⟨*proof*⟩

lemma ‹$((\forall x.\ \neg\ P(x)) \longrightarrow Q) \longrightarrow \neg\ (\forall x.\ \neg\ (P(x) \vee Q))$›

⟨*proof*⟩

lemma ‹(∀ *x*. *P*(*x*)) ∨ *Q* ⟶ (∀ *x*. *P*(*x*) ∨ *Q*)›
⟨*proof*⟩

lemma ‹(∃ *x*. *P* ⟶ *Q*(*x*)) ⟶ (*P* ⟶ (∃ *x*. *Q*(*x*)))›
⟨*proof*⟩

8.2 The following are not constructively valid!

The attempt to prove them terminates quickly!

lemma ‹((∀ *x*. *P*(*x*)) ⟶ *Q*) ⟶ (∃ *x*. *P*(*x*) ⟶ *Q*)›
⟨*proof*⟩

lemma ‹(*P* ⟶ (∃ *x*. *Q*(*x*))) ⟶ (∃ *x*. *P* ⟶ *Q*(*x*))›
⟨*proof*⟩

lemma ‹(∀ *x*. *P*(*x*) ∨ *Q*) ⟶ ((∀ *x*. *P*(*x*)) ∨ *Q*)›
⟨*proof*⟩

lemma ‹(∀ *x*. ¬ ¬ *P*(*x*)) ⟶ ¬ ¬ (∀ *x*. *P*(*x*))›
⟨*proof*⟩

Classically but not intuitionistically valid. Proved by a bug in 1986!

lemma ‹∃ *x*. *Q*(*x*) ⟶ (∀ *x*. *Q*(*x*))›
⟨*proof*⟩

8.3 Hard examples with quantifiers

The ones that have not been proved are not known to be valid! Some will require quantifier duplication – not currently available.

¬¬18

lemma ‹¬ ¬ (∃ *y*. ∀ *x*. *P*(*y*) ⟶ *P*(*x*))›
⟨*proof*⟩

¬¬19

lemma ‹¬ ¬ (∃ *x*. ∀ *y z*. (*P*(*y*) ⟶ *Q*(*z*)) ⟶ (*P*(*x*) ⟶ *Q*(*x*)))›
⟨*proof*⟩

20

lemma
‹(∀ *x y*. ∃ *z*. ∀ *w*. (*P*(*x*) ∧ *Q*(*y*) ⟶ *R*(*z*) ∧ *S*(*w*)))
⟶ (∃ *x y*. *P*(*x*) ∧ *Q*(*y*)) ⟶ (∃ *z*. *R*(*z*))›
⟨*proof*⟩

21

lemma ‹$(\exists\, x.\ P \longrightarrow Q(x)) \wedge (\exists\, x.\ Q(x) \longrightarrow P) \longrightarrow \neg\ \neg\ (\exists\, x.\ P \longleftrightarrow Q(x))$›
⟨*proof*⟩

22

lemma ‹$(\forall\, x.\ P \longleftrightarrow Q(x)) \longrightarrow (P \longleftrightarrow (\forall\, x.\ Q(x)))$›
⟨*proof*⟩

¬¬23

lemma ‹$\neg\ \neg\ ((\forall\, x.\ P \vee Q(x)) \longleftrightarrow (P \vee (\forall\, x.\ Q(x))))$›
⟨*proof*⟩

24

lemma
‹$\neg\ (\exists\, x.\ S(x) \wedge Q(x)) \wedge (\forall\, x.\ P(x) \longrightarrow Q(x) \vee R(x)) \wedge$
$(\neg\ (\exists\, x.\ P(x)) \longrightarrow (\exists\, x.\ Q(x))) \wedge (\forall\, x.\ Q(x) \vee R(x) \longrightarrow S(x))$
$\longrightarrow \neg\ \neg\ (\exists\, x.\ P(x) \wedge R(x))$›

Not clear why *fast-tac*, *best-tac*, *ASTAR* and *ITER-DEEPEN* all take forever.

⟨*proof*⟩

25

lemma
‹$(\exists\, x.\ P(x)) \wedge$
$(\forall\, x.\ L(x) \longrightarrow \neg\ (M(x) \wedge R(x))) \wedge$
$(\forall\, x.\ P(x) \longrightarrow (M(x) \wedge L(x))) \wedge$
$((\forall\, x.\ P(x) \longrightarrow Q(x)) \vee (\exists\, x.\ P(x) \wedge R(x)))$
$\longrightarrow (\exists\, x.\ Q(x) \wedge P(x))$›
⟨*proof*⟩

¬¬26

lemma
‹$(\neg\ \neg\ (\exists\, x.\ p(x)) \longleftrightarrow \neg\ \neg\ (\exists\, x.\ q(x))) \wedge$
$(\forall\, x.\ \forall\, y.\ p(x) \wedge q(y) \longrightarrow (r(x) \longleftrightarrow s(y)))$
$\longrightarrow ((\forall\, x.\ p(x) \longrightarrow r(x)) \longleftrightarrow (\forall\, x.\ q(x) \longrightarrow s(x)))$›
⟨*proof*⟩

27

lemma
‹$(\exists\, x.\ P(x) \wedge \neg\ Q(x)) \wedge$
$(\forall\, x.\ P(x) \longrightarrow R(x)) \wedge$
$(\forall\, x.\ M(x) \wedge L(x) \longrightarrow P(x)) \wedge$
$((\exists\, x.\ R(x) \wedge \neg\ Q(x)) \longrightarrow (\forall\, x.\ L(x) \longrightarrow \neg\ R(x)))$
$\longrightarrow (\forall\, x.\ M(x) \longrightarrow \neg\ L(x))$›
⟨*proof*⟩

¬¬28. AMENDED

lemma

‹(∀ x. P(x) ⟶ (∀ x. Q(x))) ∧
  (¬ ¬ (∀ x. Q(x) ∨ R(x)) ⟶ (∃ x. Q(x) ∧ S(x))) ∧
  (¬ ¬ (∃ x. S(x)) ⟶ (∀ x. L(x) ⟶ M(x)))
 ⟶ (∀ x. P(x) ∧ L(x) ⟶ M(x))›
⟨proof⟩

29. Essentially the same as Principia Mathematica *11.71

lemma
‹(∃ x. P(x)) ∧ (∃ y. Q(y))
 ⟶ ((∀ x. P(x) ⟶ R(x)) ∧ (∀ y. Q(y) ⟶ S(y)) ⟷
  (∀ x y. P(x) ∧ Q(y) ⟶ R(x) ∧ S(y)))›
⟨proof⟩

¬¬30

lemma
‹(∀ x. (P(x) ∨ Q(x)) ⟶ ¬ R(x)) ∧
  (∀ x. (Q(x) ⟶ ¬ S(x)) ⟶ P(x) ∧ R(x))
 ⟶ (∀ x. ¬ ¬ S(x))›
⟨proof⟩

31

lemma
‹¬ (∃ x. P(x) ∧ (Q(x) ∨ R(x))) ∧
  (∃ x. L(x) ∧ P(x)) ∧
  (∀ x. ¬ R(x) ⟶ M(x))
⟶ (∃ x. L(x) ∧ M(x))›
⟨proof⟩

32

lemma
‹(∀ x. P(x) ∧ (Q(x) ∨ R(x)) ⟶ S(x)) ∧
 (∀ x. S(x) ∧ R(x) ⟶ L(x)) ∧
 (∀ x. M(x) ⟶ R(x))
⟶ (∀ x. P(x) ∧ M(x) ⟶ L(x))›
⟨proof⟩

¬¬33

lemma
‹(∀ x. ¬ ¬ (P(a) ∧ (P(x) ⟶ P(b)) ⟶ P(c))) ⟷
 (∀ x. ¬ ¬ ((¬ P(a) ∨ P(x) ∨ P(c)) ∧ (¬ P(a) ∨ ¬ P(b) ∨ P(c))))›
⟨proof⟩

36

lemma
‹(∀ x. ∃ y. J(x,y)) ∧
 (∀ x. ∃ y. G(x,y)) ∧
 (∀ x y. J(x,y) ∨ G(x,y) ⟶ (∀ z. J(y,z) ∨ G(y,z) ⟶ H(x,z)))
⟶ (∀ x. ∃ y. H(x,y))›

⟨*proof*⟩

37

lemma
‹(∀ *z*. ∃ *w*. ∀ *x*. ∃ *y*.
¬ ¬ (*P*(*x*,*z*) ⟶ *P*(*y*,*w*)) ∧ *P*(*y*,*z*) ∧ (*P*(*y*,*w*) ⟶ (∃ *u*. *Q*(*u*,*w*)))) ∧
(∀ *x z*. ¬ *P*(*x*,*z*) ⟶ (∃ *y*. *Q*(*y*,*z*))) ∧
(¬ ¬ (∃ *x y*. *Q*(*x*,*y*)) ⟶ (∀ *x*. *R*(*x*,*x*)))
⟶ ¬ ¬ (∀ *x*. ∃ *y*. *R*(*x*,*y*))›
⟨*proof*⟩

39

lemma ‹¬ (∃ *x*. ∀ *y*. *F*(*y*,*x*) ⟷ ¬ *F*(*y*,*y*))›
⟨*proof*⟩

40. AMENDED

lemma
‹(∃ *y*. ∀ *x*. *F*(*x*,*y*) ⟷ *F*(*x*,*x*)) ⟶
¬ (∀ *x*. ∃ *y*. ∀ *z*. *F*(*z*,*y*) ⟷ ¬ *F*(*z*,*x*))›
⟨*proof*⟩

44

lemma
‹(∀ *x*. *f*(*x*) ⟶
(∃ *y*. *g*(*y*) ∧ *h*(*x*,*y*) ∧ (∃ *y*. *g*(*y*) ∧ ¬ *h*(*x*,*y*)))) ∧
(∃ *x*. *j*(*x*) ∧ (∀ *y*. *g*(*y*) ⟶ *h*(*x*,*y*)))
⟶ (∃ *x*. *j*(*x*) ∧ ¬ *f*(*x*))›
⟨*proof*⟩

48

lemma ‹(*a* = *b* ∨ *c* = *d*) ∧ (*a* = *c* ∨ *b* = *d*) ⟶ *a* = *d* ∨ *b* = *c*›
⟨*proof*⟩

51

lemma
‹(∃ *z w*. ∀ *x y*. *P*(*x*,*y*) ⟷ (*x* = *z* ∧ *y* = *w*)) ⟶
(∃ *z*. ∀ *x*. ∃ *w*. (∀ *y*. *P*(*x*,*y*) ⟷ *y* = *w*) ⟷ *x* = *z*)›
⟨*proof*⟩

52

Almost the same as 51.

lemma
‹(∃ *z w*. ∀ *x y*. *P*(*x*,*y*) ⟷ (*x* = *z* ∧ *y* = *w*)) ⟶
(∃ *w*. ∀ *y*. ∃ *z*. (∀ *x*. *P*(*x*,*y*) ⟷ *x* = *z*) ⟷ *y* = *w*)›
⟨*proof*⟩

56

lemma ‹$(\forall x.\ (\exists y.\ P(y) \wedge x = f(y)) \longrightarrow P(x)) \longleftrightarrow (\forall x.\ P(x) \longrightarrow P(f(x)))$›
⟨*proof*⟩

57

lemma
‹$P(f(a,b),\ f(b,c)) \wedge P(f(b,c),\ f(a,c)) \wedge$
$(\forall x\ y\ z.\ P(x,y) \wedge P(y,z) \longrightarrow P(x,z)) \longrightarrow P(f(a,b),\ f(a,c))$›
⟨*proof*⟩

60

lemma ‹$\forall x.\ P(x,f(x)) \longleftrightarrow (\exists y.\ (\forall z.\ P(z,y) \longrightarrow P(z,f(x))) \wedge P(x,y))$›
⟨*proof*⟩

end

9 First-Order Logic: propositional examples (intuitionistic version)

theory *Propositional-Int*
imports *IFOL*
begin

commutative laws of $\wedge$ and $\vee$

lemma ‹$P \wedge Q \longrightarrow Q \wedge P$›
⟨*proof*⟩

lemma ‹$P \vee Q \longrightarrow Q \vee P$›
⟨*proof*⟩

associative laws of $\wedge$ and $\vee$

lemma ‹$(P \wedge Q) \wedge R \longrightarrow P \wedge (Q \wedge R)$›
⟨*proof*⟩

lemma ‹$(P \vee Q) \vee R \longrightarrow P \vee (Q \vee R)$›
⟨*proof*⟩

distributive laws of $\wedge$ and $\vee$

lemma ‹$(P \wedge Q) \vee R \longrightarrow (P \vee R) \wedge (Q \vee R)$›
⟨*proof*⟩

lemma ‹$(P \vee R) \wedge (Q \vee R) \longrightarrow (P \wedge Q) \vee R$›
⟨*proof*⟩

lemma ‹$(P \vee Q) \wedge R \longrightarrow (P \wedge R) \vee (Q \wedge R)$›
⟨*proof*⟩

lemma ‹$(P \wedge R) \vee (Q \wedge R) \longrightarrow (P \vee Q) \wedge R$›

⟨*proof*⟩

Laws involving implication

lemma ‹$(P \longrightarrow R) \wedge (Q \longrightarrow R) \longleftrightarrow (P \vee Q \longrightarrow R)$›
⟨*proof*⟩

lemma ‹$(P \wedge Q \longrightarrow R) \longleftrightarrow (P \longrightarrow (Q \longrightarrow R))$›
⟨*proof*⟩

lemma ‹$((P \longrightarrow R) \longrightarrow R) \longrightarrow ((Q \longrightarrow R) \longrightarrow R) \longrightarrow (P \wedge Q \longrightarrow R) \longrightarrow R$›
⟨*proof*⟩

lemma ‹$\neg\ (P \longrightarrow R) \longrightarrow \neg\ (Q \longrightarrow R) \longrightarrow \neg\ (P \wedge Q \longrightarrow R)$›
⟨*proof*⟩

lemma ‹$(P \longrightarrow Q \wedge R) \longleftrightarrow (P \longrightarrow Q) \wedge (P \longrightarrow R)$›
⟨*proof*⟩

Propositions-as-types

lemma ‹$P \longrightarrow (Q \longrightarrow P)$›
⟨*proof*⟩
lemma ‹$(P \longrightarrow Q \longrightarrow R) \longrightarrow (P \longrightarrow Q) \longrightarrow (P \longrightarrow R)$›
⟨*proof*⟩
lemma ‹$(P \longrightarrow Q) \vee (P \longrightarrow R) \longrightarrow (P \longrightarrow Q \vee R)$›
⟨*proof*⟩

lemma ‹$(P \longrightarrow Q) \longrightarrow (\neg\ Q \longrightarrow \neg\ P)$›
⟨*proof*⟩

Schwichtenberg's examples (via T. Nipkow)

lemma *stab-imp*: ‹$(((Q \longrightarrow R) \longrightarrow R) \longrightarrow Q) \longrightarrow (((P \longrightarrow Q) \longrightarrow R) \longrightarrow R) \longrightarrow P \longrightarrow Q$›
⟨*proof*⟩

lemma *stab-to-peirce*:
‹$(((P \longrightarrow R) \longrightarrow R) \longrightarrow P) \longrightarrow (((Q \longrightarrow R) \longrightarrow R) \longrightarrow Q) \longrightarrow ((P \longrightarrow Q) \longrightarrow P) \longrightarrow P$›
⟨*proof*⟩

lemma *peirce-imp1*:
‹$(((Q \longrightarrow R) \longrightarrow Q) \longrightarrow Q) \longrightarrow (((P \longrightarrow Q) \longrightarrow R) \longrightarrow P \longrightarrow Q) \longrightarrow P \longrightarrow Q$›
⟨*proof*⟩

lemma *peirce-imp2*: ‹$(((P \longrightarrow R) \longrightarrow P) \longrightarrow P) \longrightarrow ((P \longrightarrow Q \longrightarrow R) \longrightarrow P) \longrightarrow P$›
⟨*proof*⟩

lemma *mints*: ‹$((((P \longrightarrow Q) \longrightarrow P) \longrightarrow P) \longrightarrow Q) \longrightarrow Q$›

⟨*proof*⟩

lemma *mints-solovev*: ‹(*P* ⟶ (*Q* ⟶ *R*) ⟶ *Q*) ⟶ ((*P* ⟶ *Q*) ⟶ *R*) ⟶ *R*›
⟨*proof*⟩

lemma *tatsuta*:
‹(((*P7* ⟶ *P1*) ⟶ *P10*) ⟶ *P4* ⟶ *P5*)
⟶ (((*P8* ⟶ *P2*) ⟶ *P9*) ⟶ *P3* ⟶ *P10*)
⟶ (*P1* ⟶ *P8*) ⟶ *P6* ⟶ *P7*
⟶ (((*P3* ⟶ *P2*) ⟶ *P9*) ⟶ *P4*)
⟶ (*P1* ⟶ *P3*) ⟶ (((*P6* ⟶ *P1*) ⟶ *P2*) ⟶ *P9*) ⟶ *P5*›
⟨*proof*⟩

lemma *tatsuta1*:
‹(((*P8* ⟶ *P2*) ⟶ *P9*) ⟶ *P3* ⟶ *P10*)
⟶ (((*P3* ⟶ *P2*) ⟶ *P9*) ⟶ *P4*)
⟶ (((*P6* ⟶ *P1*) ⟶ *P2*) ⟶ *P9*)
⟶ (((*P7* ⟶ *P1*) ⟶ *P10*) ⟶ *P4* ⟶ *P5*)
⟶ (*P1* ⟶ *P3*) ⟶ (*P1* ⟶ *P8*) ⟶ *P6* ⟶ *P7* ⟶ *P5*›
⟨*proof*⟩

end

10 First-Order Logic: quantifier examples (intuitionistic version)

theory *Quantifiers-Int*
imports *IFOL*
begin

lemma ‹(∀*x y*. *P*(*x*,*y*)) ⟶ (∀*y x*. *P*(*x*,*y*))›
⟨*proof*⟩

lemma ‹(∃*x y*. *P*(*x*,*y*)) ⟶ (∃*y x*. *P*(*x*,*y*))›
⟨*proof*⟩
lemma ‹(∀*x*. *P*(*x*)) ∨ (∀*x*. *Q*(*x*)) ⟶ (∀*x*. *P*(*x*) ∨ *Q*(*x*))›
⟨*proof*⟩

lemma ‹(∀*x*. *P* ⟶ *Q*(*x*)) ⟷ (*P* ⟶ (∀*x*. *Q*(*x*)))›
⟨*proof*⟩

lemma ‹(∀*x*. *P*(*x*) ⟶ *Q*) ⟷ ((∃*x*. *P*(*x*)) ⟶ *Q*)›
⟨*proof*⟩

Some harder ones

lemma ‹(∃*x*. *P*(*x*) ∨ *Q*(*x*)) ⟷ (∃*x*. *P*(*x*)) ∨ (∃*x*. *Q*(*x*))›
⟨*proof*⟩

lemma ‹$(\exists x.\ P(x) \wedge Q(x)) \longrightarrow (\exists x.\ P(x)) \wedge (\exists x.\ Q(x))$›
⟨*proof*⟩

Basic test of quantifier reasoning

lemma ‹$(\exists y.\ \forall x.\ Q(x,y)) \longrightarrow (\forall x.\ \exists y.\ Q(x,y))$›
⟨*proof*⟩

lemma ‹$(\forall x.\ Q(x)) \longrightarrow (\exists x.\ Q(x))$›
⟨*proof*⟩

The following should fail, as they are false!

lemma ‹$(\forall x.\ \exists y.\ Q(x,y)) \longrightarrow (\exists y.\ \forall x.\ Q(x,y))$›
⟨*proof*⟩

lemma ‹$(\exists x.\ Q(x)) \longrightarrow (\forall x.\ Q(x))$›
⟨*proof*⟩

schematic-goal ‹$P(?a) \longrightarrow (\forall x.\ P(x))$›
⟨*proof*⟩

schematic-goal ‹$(P(?a) \longrightarrow (\forall x.\ Q(x))) \longrightarrow (\forall x.\ P(x) \longrightarrow Q(x))$›
⟨*proof*⟩

Back to things that are provable ...

lemma ‹$(\forall x.\ P(x) \longrightarrow Q(x)) \wedge (\exists x.\ P(x)) \longrightarrow (\exists x.\ Q(x))$›
⟨*proof*⟩
lemma ‹$(P \longrightarrow (\exists x.\ Q(x))) \wedge P \longrightarrow (\exists x.\ Q(x))$›
⟨*proof*⟩

schematic-goal ‹$(\forall x.\ P(x) \longrightarrow Q(f(x))) \wedge (\forall x.\ Q(x) \longrightarrow R(g(x))) \wedge P(d) \longrightarrow R(?a)$›
⟨*proof*⟩

lemma ‹$(\forall x.\ Q(x)) \longrightarrow (\exists x.\ Q(x))$›
⟨*proof*⟩

Some slow ones

lemma ‹$(\forall x\ y.\ P(x) \longrightarrow Q(y)) \longleftrightarrow ((\exists x.\ P(x)) \longrightarrow (\forall y.\ Q(y)))$›
⟨*proof*⟩

lemma ‹$(\exists x\ y.\ P(x) \wedge Q(x,y)) \longleftrightarrow (\exists x.\ P(x) \wedge (\exists y.\ Q(x,y)))$›
⟨*proof*⟩

lemma ‹$(\exists y.\ \forall x.\ P(x) \longrightarrow Q(x,y)) \longrightarrow (\forall x.\ P(x) \longrightarrow (\exists y.\ Q(x,y)))$›
⟨*proof*⟩

end

11 Classical Predicate Calculus Problems

theory *Classical*
imports *FOL*
begin

lemma ‹$(P \longrightarrow Q \vee R) \longrightarrow (P \longrightarrow Q) \vee (P \longrightarrow R)$›
⟨*proof*⟩

11.0.1 If and only if

lemma ‹$(P \longleftrightarrow Q) \longleftrightarrow (Q \longleftrightarrow P)$›
⟨*proof*⟩

lemma ‹$\neg\ (P \longleftrightarrow \neg\ P)$›
⟨*proof*⟩

11.1 Pelletier's examples

Sample problems from

- F. J. Pelletier, Seventy-Five Problems for Testing Automatic Theorem Provers, J. Automated Reasoning 2 (1986), 191-216. Errata, JAR 4 (1988), 236-236.

The hardest problems – judging by experience with several theorem provers, including matrix ones – are 34 and 43.

1

lemma ‹$(P \longrightarrow Q) \longleftrightarrow (\neg\ Q \longrightarrow \neg\ P)$›
⟨*proof*⟩

2

lemma ‹$\neg\ \neg\ P \longleftrightarrow P$›
⟨*proof*⟩

3

lemma ‹$\neg\ (P \longrightarrow Q) \longrightarrow (Q \longrightarrow P)$›
⟨*proof*⟩

4

lemma ‹$(\neg\ P \longrightarrow Q) \longleftrightarrow (\neg\ Q \longrightarrow P)$›
⟨*proof*⟩

5

lemma ‹$((P \vee Q) \longrightarrow (P \vee R)) \longrightarrow (P \vee (Q \longrightarrow R))$›
⟨*proof*⟩

6

lemma ‹$P \vee \neg P$›
⟨*proof*⟩

7

lemma ‹$P \vee \neg \neg \neg P$›
⟨*proof*⟩

8. Peirce's law

lemma ‹$((P \longrightarrow Q) \longrightarrow P) \longrightarrow P$›
⟨*proof*⟩

9

lemma ‹$((P \vee Q) \wedge (\neg P \vee Q) \wedge (P \vee \neg Q)) \longrightarrow \neg (\neg P \vee \neg Q)$›
⟨*proof*⟩

10

lemma ‹$(Q \longrightarrow R) \wedge (R \longrightarrow P \wedge Q) \wedge (P \longrightarrow Q \vee R) \longrightarrow (P \longleftrightarrow Q)$›
⟨*proof*⟩

11. Proved in each direction (incorrectly, says Pelletier!!)

lemma ‹$P \longleftrightarrow P$›
⟨*proof*⟩

12. "Dijkstra's law"

lemma ‹$((P \longleftrightarrow Q) \longleftrightarrow R) \longleftrightarrow (P \longleftrightarrow (Q \longleftrightarrow R))$›
⟨*proof*⟩

13. Distributive law

lemma ‹$P \vee (Q \wedge R) \longleftrightarrow (P \vee Q) \wedge (P \vee R)$›
⟨*proof*⟩

14

lemma ‹$(P \longleftrightarrow Q) \longleftrightarrow ((Q \vee \neg P) \wedge (\neg Q \vee P))$›
⟨*proof*⟩

15

lemma ‹$(P \longrightarrow Q) \longleftrightarrow (\neg P \vee Q)$›
⟨*proof*⟩

16

lemma ‹$(P \longrightarrow Q) \vee (Q \longrightarrow P)$›
⟨*proof*⟩

17

lemma ‹$((P \wedge (Q \longrightarrow R)) \longrightarrow S) \longleftrightarrow ((\neg P \vee Q \vee S) \wedge (\neg P \vee \neg R \vee S))$›
⟨*proof*⟩

11.2 Classical Logic: examples with quantifiers

lemma ‹$(\forall x.\ P(x) \wedge Q(x)) \longleftrightarrow (\forall x.\ P(x)) \wedge (\forall x.\ Q(x))$›
⟨*proof*⟩

lemma ‹$(\exists x.\ P \longrightarrow Q(x)) \longleftrightarrow (P \longrightarrow (\exists x.\ Q(x)))$›
⟨*proof*⟩

lemma ‹$(\exists x.\ P(x) \longrightarrow Q) \longleftrightarrow (\forall x.\ P(x)) \longrightarrow Q$›
⟨*proof*⟩

lemma ‹$(\forall x.\ P(x)) \vee Q \longleftrightarrow (\forall x.\ P(x) \vee Q)$›
⟨*proof*⟩

Discussed in Avron, Gentzen-Type Systems, Resolution and Tableaux, JAR 10 (265-281), 1993. Proof is trivial!

lemma ‹$\neg\ ((\exists x.\ \neg\ P(x)) \wedge ((\exists x.\ P(x)) \vee (\exists x.\ P(x) \wedge Q(x))) \wedge \neg\ (\exists x.\ P(x)))$›
⟨*proof*⟩

11.3 Problems requiring quantifier duplication

Theorem B of Peter Andrews, Theorem Proving via General Matings, JACM 28 (1981).

lemma ‹$(\exists x.\ \forall y.\ P(x) \longleftrightarrow P(y)) \longrightarrow ((\exists x.\ P(x)) \longleftrightarrow (\forall y.\ P(y)))$›
⟨*proof*⟩

Needs multiple instantiation of ALL.

lemma ‹$(\forall x.\ P(x) \longrightarrow P(f(x))) \wedge P(d) \longrightarrow P(f(f(f(d))))$›
⟨*proof*⟩

Needs double instantiation of the quantifier

lemma ‹$\exists x.\ P(x) \longrightarrow P(a) \wedge P(b)$›
⟨*proof*⟩

lemma ‹$\exists z.\ P(z) \longrightarrow (\forall x.\ P(x))$›
⟨*proof*⟩

lemma ‹$\exists x.\ (\exists y.\ P(y)) \longrightarrow P(x)$›
⟨*proof*⟩

V. Lifschitz, What Is the Inverse Method?, JAR 5 (1989), 1–23. NOT PROVED.

lemma
‹$\exists x\ x'.\ \forall y.\ \exists z\ z'.$
$(\neg\ P(y,y) \vee P(x,x) \vee \neg\ S(z,x)) \wedge$
$(S(x,y) \vee \neg\ S(y,z) \vee Q(z',z')) \wedge$
$(Q(x',y) \vee \neg\ Q(y,z') \vee S(x',x'))$›
⟨*proof*⟩

11.4 Hard examples with quantifiers

18

lemma ‹$\exists\, y.\ \forall\, x.\ P(y) \longrightarrow P(x)$›
⟨*proof*⟩

19

lemma ‹$\exists\, x.\ \forall\, y\ z.\ (P(y) \longrightarrow Q(z)) \longrightarrow (P(x) \longrightarrow Q(x))$›
⟨*proof*⟩

20

lemma ‹$(\forall\, x\ y.\ \exists\, z.\ \forall\, w.\ (P(x) \wedge Q(y) \longrightarrow R(z) \wedge S(w)))$
$\longrightarrow (\exists\, x\ y.\ P(x) \wedge Q(y)) \longrightarrow (\exists\, z.\ R(z))$›
⟨*proof*⟩

21

lemma ‹$(\exists\, x.\ P \longrightarrow Q(x)) \wedge (\exists\, x.\ Q(x) \longrightarrow P) \longrightarrow (\exists\, x.\ P \longleftrightarrow Q(x))$›
⟨*proof*⟩

22

lemma ‹$(\forall\, x.\ P \longleftrightarrow Q(x)) \longrightarrow (P \longleftrightarrow (\forall\, x.\ Q(x)))$›
⟨*proof*⟩

23

lemma ‹$(\forall\, x.\ P \vee Q(x)) \longleftrightarrow (P \vee (\forall\, x.\ Q(x)))$›
⟨*proof*⟩

24

lemma
‹$\neg\ (\exists\, x.\ S(x) \wedge Q(x)) \wedge (\forall\, x.\ P(x) \longrightarrow Q(x) \vee R(x)) \wedge$
$(\neg\ (\exists\, x.\ P(x)) \longrightarrow (\exists\, x.\ Q(x))) \wedge (\forall\, x.\ Q(x) \vee R(x) \longrightarrow S(x))$
$\longrightarrow (\exists\, x.\ P(x) \wedge R(x))$›
⟨*proof*⟩

25

lemma
‹$(\exists\, x.\ P(x)) \wedge$
$(\forall\, x.\ L(x) \longrightarrow \neg\ (M(x) \wedge R(x))) \wedge$
$(\forall\, x.\ P(x) \longrightarrow (M(x) \wedge L(x))) \wedge$
$((\forall\, x.\ P(x) \longrightarrow Q(x)) \vee (\exists\, x.\ P(x) \wedge R(x)))$
$\longrightarrow (\exists\, x.\ Q(x) \wedge P(x))$›
⟨*proof*⟩

26

lemma
‹$((\exists\, x.\ p(x)) \longleftrightarrow (\exists\, x.\ q(x))) \wedge$
$(\forall\, x.\ \forall\, y.\ p(x) \wedge q(y) \longrightarrow (r(x) \longleftrightarrow s(y)))$

$\longrightarrow ((\forall x.\ p(x) \longrightarrow r(x)) \longleftrightarrow (\forall x.\ q(x) \longrightarrow s(x)))\rangle$
⟨*proof*⟩

27

lemma
$\langle(\exists x.\ P(x) \wedge \neg\ Q(x)) \wedge$
$(\forall x.\ P(x) \longrightarrow R(x)) \wedge$
$(\forall x.\ M(x) \wedge L(x) \longrightarrow P(x)) \wedge$
$((\exists x.\ R(x) \wedge \neg\ Q(x)) \longrightarrow (\forall x.\ L(x) \longrightarrow \neg\ R(x)))$
$\longrightarrow (\forall x.\ M(x) \longrightarrow \neg\ L(x))\rangle$
⟨*proof*⟩

28. AMENDED

lemma
$\langle(\forall x.\ P(x) \longrightarrow (\forall x.\ Q(x))) \wedge$
$((\forall x.\ Q(x) \vee R(x)) \longrightarrow (\exists x.\ Q(x) \wedge S(x))) \wedge$
$((\exists x.\ S(x)) \longrightarrow (\forall x.\ L(x) \longrightarrow M(x)))$
$\longrightarrow (\forall x.\ P(x) \wedge L(x) \longrightarrow M(x))\rangle$
⟨*proof*⟩

29. Essentially the same as Principia Mathematica *11.71

lemma
$\langle(\exists x.\ P(x)) \wedge (\exists y.\ Q(y))$
$\longrightarrow ((\forall x.\ P(x) \longrightarrow R(x)) \wedge (\forall y.\ Q(y) \longrightarrow S(y)) \longleftrightarrow$
$(\forall x\ y.\ P(x) \wedge Q(y) \longrightarrow R(x) \wedge S(y)))\rangle$
⟨*proof*⟩

30

lemma
$\langle(\forall x.\ P(x) \vee Q(x) \longrightarrow \neg\ R(x)) \wedge$
$(\forall x.\ (Q(x) \longrightarrow \neg\ S(x)) \longrightarrow P(x) \wedge R(x))$
$\longrightarrow (\forall x.\ S(x))\rangle$
⟨*proof*⟩

31

lemma
$\langle\neg\ (\exists x.\ P(x) \wedge (Q(x) \vee R(x))) \wedge$
$(\exists x.\ L(x) \wedge P(x)) \wedge$
$(\forall x.\ \neg\ R(x) \longrightarrow M(x))$
$\longrightarrow (\exists x.\ L(x) \wedge M(x))\rangle$
⟨*proof*⟩

32

lemma
$\langle(\forall x.\ P(x) \wedge (Q(x) \vee R(x)) \longrightarrow S(x)) \wedge$
$(\forall x.\ S(x) \wedge R(x) \longrightarrow L(x)) \wedge$
$(\forall x.\ M(x) \longrightarrow R(x))$
$\longrightarrow (\forall x.\ P(x) \wedge M(x) \longrightarrow L(x))\rangle$

⟨*proof*⟩

33

lemma
‹(∀ *x*. *P*(*a*) ∧ (*P*(*x*) ⟶ *P*(*b*)) ⟶ *P*(*c*)) ⟷
(∀ *x*. (¬ *P*(*a*) ∨ *P*(*x*) ∨ *P*(*c*)) ∧ (¬ *P*(*a*) ∨ ¬ *P*(*b*) ∨ *P*(*c*)))›
⟨*proof*⟩

34. AMENDED (TWICE!!). Andrews's challenge.

lemma
‹((∃ *x*. ∀ *y*. *p*(*x*) ⟷ *p*(*y*)) ⟷ ((∃ *x*. *q*(*x*)) ⟷ (∀ *y*. *p*(*y*)))) ⟷
((∃ *x*. ∀ *y*. *q*(*x*) ⟷ *q*(*y*)) ⟷ ((∃ *x*. *p*(*x*)) ⟷ (∀ *y*. *q*(*y*))))›
⟨*proof*⟩

35

lemma ‹∃ *x y*. *P*(*x*,*y*) ⟶ (∀ *u v*. *P*(*u*,*v*))›
⟨*proof*⟩

36

lemma
‹(∀ *x*. ∃ *y*. *J*(*x*,*y*)) ∧
(∀ *x*. ∃ *y*. *G*(*x*,*y*)) ∧
(∀ *x y*. *J*(*x*,*y*) ∨ *G*(*x*,*y*) ⟶ (∀ *z*. *J*(*y*,*z*) ∨ *G*(*y*,*z*) ⟶ *H*(*x*,*z*)))
⟶ (∀ *x*. ∃ *y*. *H*(*x*,*y*))›
⟨*proof*⟩

37

lemma
‹(∀ *z*. ∃ *w*. ∀ *x*. ∃ *y*.
(*P*(*x*,*z*) ⟶ *P*(*y*,*w*)) ∧ *P*(*y*,*z*) ∧ (*P*(*y*,*w*) ⟶ (∃ *u*. *Q*(*u*,*w*)))) ∧
(∀ *x z*. ¬ *P*(*x*,*z*) ⟶ (∃ *y*. *Q*(*y*,*z*))) ∧
((∃ *x y*. *Q*(*x*,*y*)) ⟶ (∀ *x*. *R*(*x*,*x*)))
⟶ (∀ *x*. ∃ *y*. *R*(*x*,*y*))›
⟨*proof*⟩

38

lemma
‹(∀ *x*. *p*(*a*) ∧ (*p*(*x*) ⟶ (∃ *y*. *p*(*y*) ∧ *r*(*x*,*y*))) ⟶
(∃ *z*. ∃ *w*. *p*(*z*) ∧ *r*(*x*,*w*) ∧ *r*(*w*,*z*))) ⟷
(∀ *x*. (¬ *p*(*a*) ∨ *p*(*x*) ∨ (∃ *z*. ∃ *w*. *p*(*z*) ∧ *r*(*x*,*w*) ∧ *r*(*w*,*z*))) ∧
(¬ *p*(*a*) ∨ ¬ (∃ *y*. *p*(*y*) ∧ *r*(*x*,*y*)) ∨
(∃ *z*. ∃ *w*. *p*(*z*) ∧ *r*(*x*,*w*) ∧ *r*(*w*,*z*))))›
⟨*proof*⟩

39

lemma ‹¬ (∃ *x*. ∀ *y*. *F*(*y*,*x*) ⟷ ¬ *F*(*y*,*y*))›
⟨*proof*⟩

40. AMENDED

lemma
‹(∃ y. ∀ x. F(x,y) ⟷ F(x,x)) ⟶
¬ (∀ x. ∃ y. ∀ z. F(z,y) ⟷ ¬ F(z,x))›
⟨*proof*⟩

41

lemma
‹(∀ z. ∃ y. ∀ x. f(x,y) ⟷ f(x,z) ∧ ¬ f(x,x))
⟶ ¬ (∃ z. ∀ x. f(x,z))›
⟨*proof*⟩

42

lemma ‹¬ (∃ y. ∀ x. p(x,y) ⟷ ¬ (∃ z. p(x,z) ∧ p(z,x)))›
⟨*proof*⟩

43

lemma
‹(∀ x. ∀ y. q(x,y) ⟷ (∀ z. p(z,x) ⟷ p(z,y)))
⟶ (∀ x. ∀ y. q(x,y) ⟷ q(y,x))›
⟨*proof*⟩

Other proofs: Can use *auto*, which cheats by using rewriting! *Deepen-tac* alone requires 253 secs. Or *by* (*mini-tac 1 THEN Deepen-tac 5 1*).

44

lemma
‹(∀ x. f(x) ⟶ (∃ y. g(y) ∧ h(x,y) ∧ (∃ y. g(y) ∧ ¬ h(x,y)))) ∧
(∃ x. j(x) ∧ (∀ y. g(y) ⟶ h(x,y)))
⟶ (∃ x. j(x) ∧ ¬ f(x))›
⟨*proof*⟩

45

lemma
‹(∀ x. f(x) ∧ (∀ y. g(y) ∧ h(x,y) ⟶ j(x,y))
⟶ (∀ y. g(y) ∧ h(x,y) ⟶ k(y))) ∧
¬ (∃ y. l(y) ∧ k(y)) ∧
(∃ x. f(x) ∧ (∀ y. h(x,y) ⟶ l(y)) ∧ (∀ y. g(y) ∧ h(x,y) ⟶ j(x,y)))
⟶ (∃ x. f(x) ∧ ¬ (∃ y. g(y) ∧ h(x,y)))›
⟨*proof*⟩

46

lemma
‹(∀ x. f(x) ∧ (∀ y. f(y) ∧ h(y,x) ⟶ g(y)) ⟶ g(x)) ∧
((∃ x. f(x) ∧ ¬ g(x)) ⟶
(∃ x. f(x) ∧ ¬ g(x) ∧ (∀ y. f(y) ∧ ¬ g(y) ⟶ j(x,y)))) ∧
(∀ x y. f(x) ∧ f(y) ∧ h(x,y) ⟶ ¬ j(y,x))
⟶ (∀ x. f(x) ⟶ g(x))›
⟨*proof*⟩

11.5 Problems (mainly) involving equality or functions

48

lemma ‹$(a = b \vee c = d) \wedge (a = c \vee b = d) \longrightarrow a = d \vee b = c$›
⟨*proof*⟩

49. NOT PROVED AUTOMATICALLY. Hard because it involves substitution for Vars; the type constraint ensures that x,y,z have the same type as a,b,u.

lemma
‹$(\exists x\ y::'a.\ \forall z.\ z = x \vee z = y) \wedge P(a) \wedge P(b) \wedge a \neq b \longrightarrow (\forall u::'a.\ P(u))$›
⟨*proof*⟩

50. (What has this to do with equality?)

lemma ‹$(\forall x.\ P(a,x) \vee (\forall y.\ P(x,y))) \longrightarrow (\exists x.\ \forall y.\ P(x,y))$›
⟨*proof*⟩

51

lemma
‹$(\exists z\ w.\ \forall x\ y.\ P(x,y) \longleftrightarrow (x = z \wedge y = w)) \longrightarrow$
$(\exists z.\ \forall x.\ \exists w.\ (\forall y.\ P(x,y) \longleftrightarrow y{=}w) \longleftrightarrow x = z)$›
⟨*proof*⟩

52

Almost the same as 51.

lemma
‹$(\exists z\ w.\ \forall x\ y.\ P(x,y) \longleftrightarrow (x = z \wedge y = w)) \longrightarrow$
$(\exists w.\ \forall y.\ \exists z.\ (\forall x.\ P(x,y) \longleftrightarrow x = z) \longleftrightarrow y = w)$›
⟨*proof*⟩

55

Non-equational version, from Manthey and Bry, CADE-9 (Springer, 1988). fast DISCOVERS who killed Agatha.

schematic-goal
‹$lives(agatha) \wedge lives(butler) \wedge lives(charles) \wedge$
$(killed(agatha,agatha) \vee killed(butler,agatha) \vee killed(charles,agatha)) \wedge$
$(\forall x\ y.\ killed(x,y) \longrightarrow hates(x,y) \wedge \neg\ richer(x,y)) \wedge$
$(\forall x.\ hates(agatha,x) \longrightarrow \neg\ hates(charles,x)) \wedge$
$(hates(agatha,agatha) \wedge hates(agatha,charles)) \wedge$
$(\forall x.\ lives(x) \wedge \neg\ richer(x,agatha) \longrightarrow hates(butler,x)) \wedge$
$(\forall x.\ hates(agatha,x) \longrightarrow hates(butler,x)) \wedge$
$(\forall x.\ \neg\ hates(x,agatha) \vee \neg\ hates(x,butler) \vee \neg\ hates(x,charles)) \longrightarrow$
$killed(?who,agatha)$›
⟨*proof*⟩

56

lemma ‹$(\forall x.\ (\exists y.\ P(y) \wedge x = f(y)) \longrightarrow P(x)) \longleftrightarrow (\forall x.\ P(x) \longrightarrow P(f(x)))$›
⟨*proof*⟩

57

lemma
‹$P(f(a,b),\ f(b,c)) \wedge P(f(b,c),\ f(a,c)) \wedge$
$(\forall x\ y\ z.\ P(x,y) \wedge P(y,z) \longrightarrow P(x,z)) \longrightarrow P(f(a,b),\ f(a,c))$›
⟨*proof*⟩

58 NOT PROVED AUTOMATICALLY

lemma ‹$(\forall x\ y.\ f(x) = g(y)) \longrightarrow (\forall x\ y.\ f(f(x)) = f(g(y)))$›
⟨*proof*⟩

59

lemma ‹$(\forall x.\ P(x) \longleftrightarrow \neg\ P(f(x))) \longrightarrow (\exists x.\ P(x) \wedge \neg\ P(f(x)))$›
⟨*proof*⟩

60

lemma ‹$\forall x.\ P(x,f(x)) \longleftrightarrow (\exists y.\ (\forall z.\ P(z,y) \longrightarrow P(z,f(x))) \wedge P(x,y))$›
⟨*proof*⟩

62 as corrected in JAR 18 (1997), page 135

lemma
‹$(\forall x.\ p(a) \wedge (p(x) \longrightarrow p(f(x))) \longrightarrow p(f(f(x)))) \longleftrightarrow$
$(\forall x.\ (\neg\ p(a) \vee p(x) \vee p(f(f(x)))) \wedge$
$(\neg\ p(a) \vee \neg\ p(f(x)) \vee p(f(f(x)))))$›
⟨*proof*⟩

From Davis, Obvious Logical Inferences, IJCAI-81, 530-531 fast indeed copes!

lemma
‹$(\forall x.\ F(x) \wedge \neg\ G(x) \longrightarrow (\exists y.\ H(x,y) \wedge J(y))) \wedge$
$(\exists x.\ K(x) \wedge F(x) \wedge (\forall y.\ H(x,y) \longrightarrow K(y))) \wedge$
$(\forall x.\ K(x) \longrightarrow \neg\ G(x)) \longrightarrow (\exists x.\ K(x) \wedge J(x))$›
⟨*proof*⟩

From Rudnicki, Obvious Inferences, JAR 3 (1987), 383-393. It does seem obvious!

lemma
‹$(\forall x.\ F(x) \wedge \neg\ G(x) \longrightarrow (\exists y.\ H(x,y) \wedge J(y))) \wedge$
$(\exists x.\ K(x) \wedge F(x) \wedge (\forall y.\ H(x,y) \longrightarrow K(y))) \wedge$
$(\forall x.\ K(x) \longrightarrow \neg\ G(x)) \longrightarrow (\exists x.\ K(x) \longrightarrow \neg\ G(x))$›
⟨*proof*⟩

Halting problem: Formulation of Li Dafa (AAR Newsletter 27, Oct 1994.) author U. Egly.

lemma
‹$((\exists x.\ A(x) \wedge (\forall y.\ C(y) \longrightarrow (\forall z.\ D(x,y,z)))) \longrightarrow$

$(\exists\, w.\ C(w) \wedge (\forall\, y.\ C(y) \longrightarrow (\forall\, z.\ D(w,y,z)))))$
$\wedge$
$(\forall\, w.\ C(w) \wedge (\forall\, u.\ C(u) \longrightarrow (\forall\, v.\ D(w,u,v))) \longrightarrow$
$(\forall\, y\ z.$
$(C(y) \wedge P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,g)) \wedge$
$(C(y) \wedge \neg\ P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,b))))$
$\wedge$
$(\forall\, w.\ C(w) \wedge$
$(\forall\, y\ z.$
$(C(y) \wedge P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,g)) \wedge$
$(C(y) \wedge \neg\ P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,b))) \longrightarrow$
$(\exists\, v.\ C(v) \wedge$
$(\forall\, y.\ ((C(y) \wedge Q(w,y,y)) \wedge OO(w,g) \longrightarrow \neg\ P(v,y)) \wedge$
$((C(y) \wedge Q(w,y,y)) \wedge OO(w,b) \longrightarrow P(v,y) \wedge OO(v,b)))))$
$\longrightarrow \neg\ (\exists\, x.\ A(x) \wedge (\forall\, y.\ C(y) \longrightarrow (\forall\, z.\ D(x,y,z))))\rangle$
⟨*proof*⟩

Halting problem II: credited to M. Bruschi by Li Dafa in JAR 18(1), p. 105.

lemma
$\langle((\exists\, x.\ A(x) \wedge (\forall\, y.\ C(y) \longrightarrow (\forall\, z.\ D(x,y,z)))) \longrightarrow$
$(\exists\, w.\ C(w) \wedge (\forall\, y.\ C(y) \longrightarrow (\forall\, z.\ D(w,y,z)))))$
$\wedge$
$(\forall\, w.\ C(w) \wedge (\forall\, u.\ C(u) \longrightarrow (\forall\, v.\ D(w,u,v))) \longrightarrow$
$(\forall\, y\ z.$
$(C(y) \wedge P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,g)) \wedge$
$(C(y) \wedge \neg\ P(y,z) \longrightarrow Q(w,y,z) \wedge OO(w,b))))$
$\wedge$
$((\exists\, w.\ C(w) \wedge (\forall\, y.\ (C(y) \wedge P(y,y) \longrightarrow Q(w,y,y) \wedge OO(w,g)) \wedge$
$(C(y) \wedge \neg\ P(y,y) \longrightarrow Q(w,y,y) \wedge OO(w,b))))$
$\longrightarrow$
$(\exists\, v.\ C(v) \wedge (\forall\, y.\ (C(y) \wedge P(y,y) \longrightarrow P(v,y) \wedge OO(v,g)) \wedge$
$(C(y) \wedge \neg\ P(y,y) \longrightarrow P(v,y) \wedge OO(v,b)))))$
$\longrightarrow$
$((\exists\, v.\ C(v) \wedge (\forall\, y.\ (C(y) \wedge P(y,y) \longrightarrow P(v,y) \wedge OO(v,g)) \wedge$
$(C(y) \wedge \neg\ P(y,y) \longrightarrow P(v,y) \wedge OO(v,b))))$
$\longrightarrow$
$(\exists\, u.\ C(u) \wedge (\forall\, y.\ (C(y) \wedge P(y,y) \longrightarrow \neg\ P(u,y)) \wedge$
$(C(y) \wedge \neg\ P(y,y) \longrightarrow P(u,y) \wedge OO(u,b)))))$
$\longrightarrow \neg\ (\exists\, x.\ A(x) \wedge (\forall\, y.\ C(y) \longrightarrow (\forall\, z.\ D(x,y,z))))\rangle$
⟨*proof*⟩

Challenge found on info-hol.

lemma $\langle\forall\, x.\ \exists\, v\ w.\ \forall\, y\ z.\ P(x) \wedge Q(y) \longrightarrow (P(v) \vee R(w)) \wedge (R(z) \longrightarrow Q(v))\rangle$
⟨*proof*⟩

Attributed to Lewis Carroll by S. G. Pulman. The first or last assumption can be deleted.

lemma
$\langle(\forall\, x.\ \mathit{honest}(x) \wedge \mathit{industrious}(x) \longrightarrow \mathit{healthy}(x)) \wedge$

$\neg\ (\exists\, x.\ grocer(x) \wedge healthy(x)) \wedge$
$(\forall\, x.\ industrious(x) \wedge grocer(x) \longrightarrow honest(x)) \wedge$
$(\forall\, x.\ cyclist(x) \longrightarrow industrious(x)) \wedge$
$(\forall\, x.\ \neg\ healthy(x) \wedge cyclist(x) \longrightarrow \neg\ honest(x))$
$\longrightarrow (\forall\, x.\ grocer(x) \longrightarrow \neg\ cyclist(x))\rangle$
⟨*proof*⟩

end

12 First-Order Logic: propositional examples (classical version)

theory *Propositional-Cla*
imports *FOL*
begin

commutative laws of $\wedge$ and $\vee$

lemma ‹$P \wedge Q \longrightarrow Q \wedge P$›
⟨*proof*⟩

lemma ‹$P \vee Q \longrightarrow Q \vee P$›
⟨*proof*⟩

associative laws of $\wedge$ and $\vee$

lemma ‹$(P \wedge Q) \wedge R \longrightarrow P \wedge (Q \wedge R)$›
⟨*proof*⟩

lemma ‹$(P \vee Q) \vee R \longrightarrow\ P \vee (Q \vee R)$›
⟨*proof*⟩

distributive laws of $\wedge$ and $\vee$

lemma ‹$(P \wedge Q) \vee R \longrightarrow (P \vee R) \wedge (Q \vee R)$›
⟨*proof*⟩

lemma ‹$(P \vee R) \wedge (Q \vee R) \longrightarrow (P \wedge Q) \vee R$›
⟨*proof*⟩

lemma ‹$(P \vee Q) \wedge R \longrightarrow (P \wedge R) \vee (Q \wedge R)$›
⟨*proof*⟩

lemma ‹$(P \wedge R) \vee (Q \wedge R) \longrightarrow (P \vee Q) \wedge R$›
⟨*proof*⟩

Laws involving implication

lemma ‹$(P \longrightarrow R) \wedge (Q \longrightarrow R) \longleftrightarrow (P \vee Q \longrightarrow R)$›

⟨*proof*⟩

lemma ‹$(P \wedge Q \longrightarrow R) \longleftrightarrow (P \longrightarrow (Q \longrightarrow R))$›
⟨*proof*⟩

lemma ‹$((P \longrightarrow R) \longrightarrow R) \longrightarrow ((Q \longrightarrow R) \longrightarrow R) \longrightarrow (P \wedge Q \longrightarrow R) \longrightarrow R$›
⟨*proof*⟩

lemma ‹$\neg\ (P \longrightarrow R) \longrightarrow \neg\ (Q \longrightarrow R) \longrightarrow \neg\ (P \wedge Q \longrightarrow R)$›
⟨*proof*⟩

lemma ‹$(P \longrightarrow Q \wedge R) \longleftrightarrow (P \longrightarrow Q) \wedge (P \longrightarrow R)$›
⟨*proof*⟩

Propositions-as-types

lemma ‹$P \longrightarrow (Q \longrightarrow P)$›
⟨*proof*⟩
lemma ‹$(P \longrightarrow Q \longrightarrow R) \longrightarrow (P \longrightarrow Q) \longrightarrow (P \longrightarrow R)$›
⟨*proof*⟩
lemma ‹$(P \longrightarrow Q) \vee (P \longrightarrow R) \longrightarrow (P \longrightarrow Q \vee R)$›
⟨*proof*⟩

lemma ‹$(P \longrightarrow Q) \longrightarrow (\neg\ Q \longrightarrow \neg\ P)$›
⟨*proof*⟩

Schwichtenberg's examples (via T. Nipkow)

lemma *stab-imp*: ‹$(((Q \longrightarrow R) \longrightarrow R) \longrightarrow Q) \longrightarrow (((P \longrightarrow Q) \longrightarrow R) \longrightarrow R) \longrightarrow P \longrightarrow Q$›
⟨*proof*⟩

lemma *stab-to-peirce*:
‹$(((P \longrightarrow R) \longrightarrow R) \longrightarrow P) \longrightarrow (((Q \longrightarrow R) \longrightarrow R) \longrightarrow Q) \longrightarrow ((P \longrightarrow Q) \longrightarrow P) \longrightarrow P$›
⟨*proof*⟩

lemma *peirce-imp1*:
‹$(((Q \longrightarrow R) \longrightarrow Q) \longrightarrow Q) \longrightarrow (((P \longrightarrow Q) \longrightarrow R) \longrightarrow P \longrightarrow Q) \longrightarrow P \longrightarrow Q$›
⟨*proof*⟩

lemma *peirce-imp2*: ‹$(((P \longrightarrow R) \longrightarrow P) \longrightarrow P) \longrightarrow ((P \longrightarrow Q \longrightarrow R) \longrightarrow P) \longrightarrow P$›
⟨*proof*⟩

lemma *mints*: ‹$((((P \longrightarrow Q) \longrightarrow P) \longrightarrow P) \longrightarrow Q) \longrightarrow Q$›
⟨*proof*⟩

lemma *mints-solovev*: ‹$(P \longrightarrow (Q \longrightarrow R) \longrightarrow Q) \longrightarrow ((P \longrightarrow Q) \longrightarrow R) \longrightarrow R$›
⟨*proof*⟩

lemma *tatsuta*:
‹(((*P7* ⟶ *P1*) ⟶ *P10*) ⟶ *P4* ⟶ *P5*)
⟶ (((*P8* ⟶ *P2*) ⟶ *P9*) ⟶ *P3* ⟶ *P10*)
⟶ (*P1* ⟶ *P8*) ⟶ *P6* ⟶ *P7*
⟶ (((*P3* ⟶ *P2*) ⟶ *P9*) ⟶ *P4*)
⟶ (*P1* ⟶ *P3*) ⟶ (((*P6* ⟶ *P1*) ⟶ *P2*) ⟶ *P9*) ⟶ *P5*›
⟨*proof*⟩

lemma *tatsuta1*:
‹(((*P8* ⟶ *P2*) ⟶ *P9*) ⟶ *P3* ⟶ *P10*)
⟶ (((*P3* ⟶ *P2*) ⟶ *P9*) ⟶ *P4*)
⟶ (((*P6* ⟶ *P1*) ⟶ *P2*) ⟶ *P9*)
⟶ (((*P7* ⟶ *P1*) ⟶ *P10*) ⟶ *P4* ⟶ *P5*)
⟶ (*P1* ⟶ *P3*) ⟶ (*P1* ⟶ *P8*) ⟶ *P6* ⟶ *P7* ⟶ *P5*›
⟨*proof*⟩

end

13 First-Order Logic: quantifier examples (classical version)

theory *Quantifiers-Cla*
imports *FOL*
begin

lemma ‹(∀*x y*. *P*(*x*,*y*)) ⟶ (∀*y x*. *P*(*x*,*y*))›
⟨*proof*⟩

lemma ‹(∃*x y*. *P*(*x*,*y*)) ⟶ (∃*y x*. *P*(*x*,*y*))›
⟨*proof*⟩

Converse is false.

lemma ‹(∀*x*. *P*(*x*)) ∨ (∀*x*. *Q*(*x*)) ⟶ (∀*x*. *P*(*x*) ∨ *Q*(*x*))›
⟨*proof*⟩

lemma ‹(∀*x*. *P* ⟶ *Q*(*x*)) ⟷ (*P* ⟶ (∀*x*. *Q*(*x*)))›
⟨*proof*⟩

lemma ‹(∀*x*. *P*(*x*) ⟶ *Q*) ⟷ ((∃*x*. *P*(*x*)) ⟶ *Q*)›
⟨*proof*⟩

Some harder ones.

lemma ‹(∃*x*. *P*(*x*) ∨ *Q*(*x*)) ⟷ (∃*x*. *P*(*x*)) ∨ (∃*x*. *Q*(*x*))›
⟨*proof*⟩
lemma ‹(∃*x*. *P*(*x*) ∧ *Q*(*x*)) ⟶ (∃*x*. *P*(*x*)) ∧ (∃*x*. *Q*(*x*))›
⟨*proof*⟩

Basic test of quantifier reasoning.

lemma ‹$(\exists\, y.\ \forall\, x.\ Q(x,y)) \longrightarrow (\forall\, x.\ \exists\, y.\ Q(x,y))$›
⟨*proof*⟩

lemma ‹$(\forall\, x.\ Q(x)) \longrightarrow (\exists\, x.\ Q(x))$›
⟨*proof*⟩

The following should fail, as they are false!

lemma ‹$(\forall\, x.\ \exists\, y.\ Q(x,y)) \longrightarrow (\exists\, y.\ \forall\, x.\ Q(x,y))$›
⟨*proof*⟩

lemma ‹$(\exists\, x.\ Q(x)) \longrightarrow (\forall\, x.\ Q(x))$›
⟨*proof*⟩

schematic-goal ‹$P(?a) \longrightarrow (\forall\, x.\ P(x))$›
⟨*proof*⟩

schematic-goal ‹$(P(?a) \longrightarrow (\forall\, x.\ Q(x))) \longrightarrow (\forall\, x.\ P(x) \longrightarrow Q(x))$›
⟨*proof*⟩

Back to things that are provable ...

lemma ‹$(\forall\, x.\ P(x) \longrightarrow Q(x)) \wedge (\exists\, x.\ P(x)) \longrightarrow (\exists\, x.\ Q(x))$›
⟨*proof*⟩

An example of why *exI* should be delayed as long as possible.

lemma ‹$(P \longrightarrow (\exists\, x.\ Q(x))) \wedge P \longrightarrow (\exists\, x.\ Q(x))$›
⟨*proof*⟩

schematic-goal ‹$(\forall\, x.\ P(x) \longrightarrow Q(f(x))) \wedge (\forall\, x.\ Q(x) \longrightarrow R(g(x))) \wedge P(d) \longrightarrow R(?a)$›
⟨*proof*⟩

lemma ‹$(\forall\, x.\ Q(x)) \longrightarrow (\exists\, x.\ Q(x))$›
⟨*proof*⟩

Some slow ones

Principia Mathematica *11.53

lemma ‹$(\forall\, x\ y.\ P(x) \longrightarrow Q(y)) \longleftrightarrow ((\exists\, x.\ P(x)) \longrightarrow (\forall\, y.\ Q(y)))$›
⟨*proof*⟩

lemma ‹$(\exists\, x\ y.\ P(x) \wedge Q(x,y)) \longleftrightarrow (\exists\, x.\ P(x) \wedge (\exists\, y.\ Q(x,y)))$›
⟨*proof*⟩

lemma ‹$(\exists\, y.\ \forall\, x.\ P(x) \longrightarrow Q(x,y)) \longrightarrow (\forall\, x.\ P(x) \longrightarrow (\exists\, y.\ Q(x,y)))$›
⟨*proof*⟩

end

theory *Miniscope*
imports *FOL*
begin

lemmas *ccontr* = *FalseE* [*THEN classical*]

13.1 Negation Normal Form

13.1.1 de Morgan laws

lemma *demorgans1*:
‹¬ (*P* ∧ *Q*) ⟷ ¬ *P* ∨ ¬ *Q*›
‹¬ (*P* ∨ *Q*) ⟷ ¬ *P* ∧ ¬ *Q*›
‹¬ ¬ *P* ⟷ *P*›
⟨*proof*⟩

lemma *demorgans2*:
‹⋀*P*. ¬ (∀ *x*. *P*(*x*)) ⟷ (∃ *x*. ¬ *P*(*x*))›
‹⋀*P*. ¬ (∃ *x*. *P*(*x*)) ⟷ (∀ *x*. ¬ *P*(*x*))›
⟨*proof*⟩

lemmas *demorgans* = *demorgans1 demorgans2*

lemma *nnf-simps*:
‹(*P* ⟶ *Q*) ⟷ (¬ *P* ∨ *Q*)›
‹¬ (*P* ⟶ *Q*) ⟷ (*P* ∧ ¬ *Q*)›
‹(*P* ⟷ *Q*) ⟷ (¬ *P* ∨ *Q*) ∧ (¬ *Q* ∨ *P*)›
‹¬ (*P* ⟷ *Q*) ⟷ (*P* ∨ *Q*) ∧ (¬ *P* ∨ ¬ *Q*)›
⟨*proof*⟩

13.1.2 Pushing in the existential quantifiers

lemma *ex-simps*:
‹(∃ *x*. *P*) ⟷ *P*›
‹⋀*P Q*. (∃ *x*. *P*(*x*) ∧ *Q*) ⟷ (∃ *x*. *P*(*x*)) ∧ *Q*›
‹⋀*P Q*. (∃ *x*. *P* ∧ *Q*(*x*)) ⟷ *P* ∧ (∃ *x*. *Q*(*x*))›
‹⋀*P Q*. (∃ *x*. *P*(*x*) ∨ *Q*(*x*)) ⟷ (∃ *x*. *P*(*x*)) ∨ (∃ *x*. *Q*(*x*))›
‹⋀*P Q*. (∃ *x*. *P*(*x*) ∨ *Q*) ⟷ (∃ *x*. *P*(*x*)) ∨ *Q*›
‹⋀*P Q*. (∃ *x*. *P* ∨ *Q*(*x*)) ⟷ *P* ∨ (∃ *x*. *Q*(*x*))›
⟨*proof*⟩

13.1.3 Pushing in the universal quantifiers

lemma *all-simps*:

‹(∀ x. P) ⟷ P›
‹⋀P Q. (∀ x. P(x) ∧ Q(x)) ⟷ (∀ x. P(x)) ∧ (∀ x. Q(x))›
‹⋀P Q. (∀ x. P(x) ∧ Q) ⟷ (∀ x. P(x)) ∧ Q›
‹⋀P Q. (∀ x. P ∧ Q(x)) ⟷ P ∧ (∀ x. Q(x))›
‹⋀P Q. (∀ x. P(x) ∨ Q) ⟷ (∀ x. P(x)) ∨ Q›
‹⋀P Q. (∀ x. P ∨ Q(x)) ⟷ P ∨ (∀ x. Q(x))›
⟨*proof*⟩

lemmas *mini-simps* = *demorgans nnf-simps ex-simps all-simps*

⟨*ML*⟩

end

14 First-Order Logic: the 'if' example

theory *If*
imports *FOL*
begin

definition *if* :: ‹[o,o,o]=>o›
  where ‹*if*(P,Q,R) ≡ P ∧ Q ∨ ¬ P ∧ R›

lemma *ifI*: ‹⟦P ⟹ Q; ¬ P ⟹ R⟧ ⟹ *if*(P,Q,R)›
  ⟨*proof*⟩

lemma *ifE*: ‹⟦*if*(P,Q,R); ⟦P; Q⟧ ⟹ S; ⟦¬ P; R⟧ ⟹ S⟧ ⟹ S›
  ⟨*proof*⟩

lemma *if-commute*: ‹*if*(P, *if*(Q,A,B), *if*(Q,C,D)) ⟷ *if*(Q, *if*(P,A,C), *if*(P,B,D))›
  ⟨*proof*⟩

Trying again from the beginning in order to use *blast*

declare *ifI* [*intro*!]
declare *ifE* [*elim*!]

lemma *if-commute*: ‹*if*(P, *if*(Q,A,B), *if*(Q,C,D)) ⟷ *if*(Q, *if*(P,A,C), *if*(P,B,D))›
  ⟨*proof*⟩

lemma ‹*if*(*if*(P,Q,R), A, B) ⟷ *if*(P, *if*(Q,A,B), *if*(R,A,B))›
  ⟨*proof*⟩

Trying again from the beginning in order to prove from the definitions

lemma ‹*if*(*if*(P,Q,R), A, B) ⟷ *if*(P, *if*(Q,A,B), *if*(R,A,B))›
  ⟨*proof*⟩

An invalid formula. High-level rules permit a simpler diagnosis.

lemma ‹$if(if(P,Q,R),\ A,\ B) \longleftrightarrow if(P,\ if(Q,A,B),\ if(R,B,A))$›
⟨*proof*⟩

Trying again from the beginning in order to prove from the definitions.

lemma ‹$if(if(P,Q,R),\ A,\ B) \longleftrightarrow if(P,\ if(Q,A,B),\ if(R,B,A))$›
⟨*proof*⟩

end