1: Operators of First-order Intuitionistic Logic

FOIL	KeYmaera I	Operator	Informal meaning
$p(e_1,\ldots,e_k)$	`p(e1,...,ek)`	predicate	arbitrary proposition may depend on $e_1,\ldots,e_k$
$\top$	`true`	truth	is always true
$\bot$	`false`	falsehood	has no proof
$\neg A$	`!A`	negation / not	A implies false
$A \land B$	`A & B`	conjunction / and	both A and B are true
$A \lor B$	`A | B`	disjunction / or	evident that A is true or evident that B is true
$A \supset B$	`A -> B`	implication / implies	truth of A implies truth of B
$A \equiv B$	`A <-> B`	bi-implication / equivalent	A implies B as well as B implies A
$\forall x\, A$	`\forall x A`	universal quantifier / for all	A for all x of (existent) type τ short scope
$\exists x\, A$	`\exists x A`	existential quantifier / exists	A for some witness for x of (existent) type τ

```
Definitions
  Bool p;        /* predicate symbol of no arguments */
  Bool q;
  Bool r(R);     /* predicate symbol of one argument */
End.

Problem
  (p|q) -> (p->\forall x r(x)) & (q->\forall x r(x))
  -> \forall x r(x)
End.
```

```
Definitions
  Bool p;        /* predicate symbol of no arguments */
  Bool q;
  Bool r;
  Bool s;
End.

Problem
  p&q -> (q->r) -> (p->(r->s)) -> s
End.

Tactic "explicit proof"
  implycR(1) ; implycR(1) ; implycR(1) ; andcL1(-1) ; andcL2(-3) ; implycL(-1) ; <(
  id,
  implycL(-2) ; <(
    id,
    implycL(-2) ; <(
      id,
      id
      )
    )
  )
End.
```

2: Intuitionistic propositional sequent calculus proof rules

andcR $\land$R $\dfrac{\Gamma \Longrightarrow A \quad \Gamma \Longrightarrow B}{\Gamma \Longrightarrow A \land B}$

orcR1 $\lor\mathrm{R}_1$ $\dfrac{\Gamma \Longrightarrow A}{\Gamma \Longrightarrow A \lor B}$

andcL1 $\land\mathrm{L}_1$ $\dfrac{\Gamma, A \land B, A \Longrightarrow C}{A \land B, \Gamma \Longrightarrow C}$

orcR2 $\lor\mathrm{R}_2$ $\dfrac{\Gamma \Longrightarrow B}{\Gamma \Longrightarrow A \lor B}$

andcL2 $\land\mathrm{L}_2$ $\dfrac{\Gamma, A \land B, B \Longrightarrow C}{A \land B, \Gamma \Longrightarrow C}$

orcL $\lor$L $\dfrac{A, \Gamma \Longrightarrow C \quad B, \Gamma \Longrightarrow C}{A \lor B, \Gamma \Longrightarrow C}$

implycR $\supset$R $\dfrac{\Gamma, A \Longrightarrow B}{\Gamma \Longrightarrow A \supset B}$

notcR $\neg$R $\dfrac{\Gamma, A \Longrightarrow \bot}{\Gamma \Longrightarrow \neg A}$

implycL $\supset$L $\dfrac{A \supset B, \Gamma \Longrightarrow A \quad B, \Gamma \Longrightarrow C}{A \supset B, \Gamma \Longrightarrow C}$

notcL $\neg$L $\dfrac{\neg A, \Gamma \Longrightarrow A \quad \bot, \Gamma \Longrightarrow C}{\neg A, \Gamma \Longrightarrow C}$

closeTrue $\top$R $\dfrac{}{\Gamma \Longrightarrow \top}$

closeFalse $\bot$L $\dfrac{}{\bot, \Gamma \Longrightarrow C}$

id init $\dfrac{}{P, \Gamma \Longrightarrow P}$

3: Quantifier sequent calculus proof rules (uni-typed with existence presupposition)

allcR $\forall$R $\dfrac{\Gamma \Longrightarrow A(c)}{\Gamma \Longrightarrow \forall x\, A(x)}$ (c new)

existscR $\exists$R $\dfrac{\Gamma \Longrightarrow A(e)}{\Gamma \Longrightarrow \exists x\, A(x)}$

allcL $\forall$L $\dfrac{\forall x\, A(x), \Gamma, A(e) \Longrightarrow C}{\forall x\, A(x), \Gamma \Longrightarrow C}$

existscL $\exists$L $\dfrac{A(c), \Gamma \Longrightarrow C}{\exists x\, A(x), \Gamma \Longrightarrow C}$ (c new)

4: Admissible rules

close id $\dfrac{}{A, \Gamma \Longrightarrow A}$

cut cut $\dfrac{\Gamma, D \Longrightarrow C \quad \Gamma \Longrightarrow D}{\Gamma \Longrightarrow C}$

hideL WL $\dfrac{\Gamma \Longrightarrow C}{A, \Gamma \Longrightarrow C}$

5: Bellerophon tactic language operators for proof search

Bellerophon	Operation	Effect
`s ; t`	sequential composition	run `t` on the output of `s`, failing if either fail
`s | t`	alternative choice	run `t` if applying `s` failed, failing if both fail
`t*`	saturating repetition	repeats tactic `t` until nothing changes any more
`t*n`	bounded repetition	repeats tactic `t` exactly `n` times, failing if any of those repetitions fail
`<(t1,...,tn)`	branching	runs tactic `ti` on branch `i`, failing if any fail or if branches ≠`n`
`t(j)`	at position	applies tactic `t` at position `j` of the sequent
`t(j,e)`	at position	applies tactic `t` to expression `e`, which is at position `j` of the sequent
`1`	succedent position	position of first succedent formula. Similar `2`, `3`, ..., `'Rlast`
`-1`	antecedent position	position of first antecedent formula. Similar `-2`, `-3`, ..., `'Llast`
`-4.0.1`	subposition	second child of first child of fourth antecedent formula Similar `4.0.1`
`'R`	search succedent	first applicable succedent position (where formula `e` is, if specified)
`'L`	search antecedent	first applicable antecedent position (where formula `e` is, if specified)
`'_`	search sequent	first applicable antecedent/succedent position (where `e` is, if specified)