Leibniz

/Reference manual/Z-related commands/In situ replacement commands

The *Leibniz* command performs rewriting by equalities and equivalences. The expression e_1 can be rewritten as the expression e_2 if the equality $e_1 = e_2$ (or the equality $e_2 = e_1$) is, in a general sense defined below, an antecedent of it. Similarly, the predicate p_1 can be rewritten as the predicate p_2 if the equivalence $p_1 \Leftrightarrow p_2$ (or the equivalence $p_2 \Leftrightarrow p_1$) is an antecedent of it.

In the case of an equality with the name of a local variable on one side, a quicker way to eliminate all uses of the variable is provided by the *one-point* command.

The simplest case of *Leibniz* to explain is that where an equality is an antecedent predicate of the entire conjecture. The e_1 must then be in either the antecedent or consequent part of the goal, as in the declaration part the equality would not be an antecedent of the e_1.

In the following rules, the postfix notation $[new/old]$ denotes the substitution of the *new* expression for the *old* expression, and the notation $a(b)$ means that b appears within a.

$$\frac{\mid e_1 = e_2, p[e_2/e_1] \vdash ?}{\mid e_1 = e_2, p(e_1) \vdash ?}$$

$$\frac{\mid e_1 = e_2 \vdash ?\ p[e_2/e_1]}{\mid e_1 = e_2 \vdash ?\ p(e_1)}$$

$$\frac{\mid e_2 = e_1, p[e_2/e_1] \vdash ?}{\mid e_2 = e_1, p(e_1) \vdash ?}$$

$$\frac{\mid e_2 = e_1 \vdash ?\ p[e_2/e_1]}{\mid e_2 = e_1 \vdash ?\ p(e_1)}$$

The expression e_1 should be selected and crossed, then the equality $e_1 = e_2$ (or $e_2 = e_1$) should be selected and inspected.

There are analogous proof rules for performing rewriting on predicates based on an equivalence as an antecedent.

$$\frac{\mid p_1 \Leftrightarrow p_2, p[p_2/p_1] \vdash ?}{\mid p_1 \Leftrightarrow p_2, p(p_1) \vdash ?}$$

$$\frac{\mid p_1 \Leftrightarrow p_2 \vdash ?\ p[p_2/p_1]}{\mid p_1 \Leftrightarrow p_2 \vdash ?\ p(p_1)}$$

$$\frac{\mid p_2 \Leftrightarrow p_1, p[p_2/p_1] \vdash ?}{\mid p_2 \Leftrightarrow p_1, p(p_1) \vdash ?}$$

$$\frac{\mid p_2 \Leftrightarrow p_1 \vdash ?\ p[p_2/p_1]}{\mid p_2 \Leftrightarrow p_1 \vdash ?\ p(p_1)}$$

The predicate p_1 should be selected and crossed, then the equivalence $p_1 \Leftrightarrow p_2$ (or $p_2 \Leftrightarrow p_1$) should be selected and inspected.

Since any predicate p_1 is equivalent to $p_1 \Leftrightarrow \mathit{true}$, there are analogous rules that rewrite predicates appearing as antecedents to *true*.

$$\frac{\mid p_1, p[\mathit{true}/p_1] \vdash ?}{\mid p_1, p(p_1) \vdash ?}$$

$$\frac{\mid p_1 \vdash ?\ p[\mathit{true}/p_1]}{\mid p_1 \vdash ?\ p(p_1)}$$

Similarly, since any predicate $\neg\, p_1$ is equivalent to $p_1 \Leftrightarrow \mathit{false}$, there are analogous rules that rewrite predicates whose negations appear as antecedents to *false*.

$$\frac{\mid \neg\, p_1, p[\mathit{false}/p_1] \vdash ?}{\mid \neg\, p_1, p(p_1) \vdash ?}$$

$$\frac{\mid \neg\, p_1 \vdash ?\ p[\mathit{false}/p_1]}{\mid \neg\, p_1 \vdash ?\ p(p_1)}$$

All of the above *Leibniz* rules concern the case where the equality or equivalence is an antecedent of the entire goal. *Leibniz* is also applicable when the equality or equivalence is more narrowly scoped, so long as it is still antecedent to the expression or predicate to be rewritten. This means that it can be a conjunct of any of a surrounding conjunction predicate, the left operand of a surrounding implication predicate, the | part of a surrounding schema text, or the • part of a surrounding (unique) existential predicate, or, as above, an antecedent predicate of the entire goal.

Several rewritings may be performed in one use of the *Leibniz* command: all the expressions or predicates to be replaced must be selected and crossed (using the previewer's `x` command), then the equality or equivalence should be inspected. The *Leibniz* command must be applicable to all the crossed selections for it to be offered, though not all the rewritings need be in the same direction.

For rewriting by quantified equalities and equivalences, see the *rewrite by antecedent* command.

1. Tactic example

"Leibniz" e_1 e_2 p_3

This example rewrites the expressions e_1 and e_2 according to the equality p_3.

IT 30-Mar-2000