iteration

This command is part of the experimental refinement editor.

The *iteration* command refines a specification statement to a loop whose body is a non-deterministic alternation. It applies the following inference rule of the refinement calculus.

$$\frac{\begin{array}{l}\vdash?\ I \wedge \neg\,(G_1 \vee ... \vee G_n) \Rightarrow Q \\ \vdash?\ I \wedge \neg\ v \in \mathbb{N} \Rightarrow \neg\,(G_1 \vee ... \vee G_n) \\ \vdash?\ \forall V : \mathbb{N} \bullet \Delta F[G_1 \wedge V = v, I, v < V] \\ ... \\ \vdash?\ \forall V : \mathbb{N} \bullet \Delta F[G_n \wedge V = v, I, v < V]\end{array}}{\vdash?\ \Delta F[P, I, Q]}$$

where V is a reference to a variable whose value is that of the variant expression and $G_1 ... G_n$ are guard predicates.

The *code* that is implicitly generated by this refinement rule is the following non-deterministic iteration of guarded specification statements.

$$\begin{array}{l}do\ G_1 \rightarrow \Delta F[G_1 \wedge P, I, Q] \\ [] \ ... \\ []\ G_n \rightarrow \Delta F[G_n \wedge P, I, Q] \\ od\end{array}$$

The *iteration* command is applicable when any specification statement $\Delta F[P, I, Q]$

in a goal is inspected.

The variant V is entered into a dialogue box using the syntax of a Z expression, and it must be of type $\mathbb{A}$. The default response is the previous response. Alternatively, if a suitable expression is displayed in the same window, it can have been selected first (crossed). Then, each guard in the sequence $G_1 ... G_n$ is entered into one of a sequence of dialogue boxes using the syntax of a Z predicate. The sequence is ended by giving an empty response. The default response is nothing. Alternatively, if some suitable predicates are displayed in the same window, they can have been selected first (crossed). Additional guards can then be entered in dialogue boxes. The variant and guards are typechecked in the environment of the inspected specification statement. An error in any dialogue box aborts the entire *iteration* command.

There must be at least one guard.

1. Tactic example

"iteration" "G_1" ... "G_n" "v" p

This example applies the *iteration* command to specification statement p using variant expression v and the guards $G_1 ... G_n$.

A tactic that applies the *iteration* command must be executed by *play tactic*; it

is not applicable under *apply tactic* as the resulting code would not be accessible to the *code* command.

IT 20-Nov-2000