

COMP 340-08B


Reasoning about Programs

ProofNavigator Reference Chart

The Basics

The RISC ProofNavigator is installed in the Linux labs and can be run using the command

ProofNavigator


ProofNavigator automatically saves every proof once it is completed. You can also save an incomplete proof: simply press the exit button () and confirm that you want to exit. When you open the same proof again, you will be asked whether you want to “*Replay the skeleton proof*”, and if you answer “*Y*” to this question, ProofNavigator will restore the state from before you exited.

Commands

Following are the most common ProofNavigator commands that you will need to complete the assignments. Most of these commands can be started from the menu, and some can also be invoked using the toolbar at the bottom of the main window.

assume $\langle formula \rangle$ — Introduces a new formula as an intermediate step. This command produces two new proof states: the first one is obtained by adding the given *formula* as an assumption to the current proof state, and in the second state this *formula* is to be proved from the current assumptions.

Example. `assume gcd(x_0,0) >= x_0 AND gcd(x_0,0) <= x_0;`

auto () — Tries to close the proof state by trying various instantiations for all universally quantified assumptions, and all existentially quantified goals. Only a small number of simple instantiations is attempted, and if they do not help to close the proof state, the command has no effect. Nevertheless, this command can take some time.


auto $\langle line \rangle$ — Same as **auto**, but tries to instantiate only the formula in the line given.

expand $\langle name \rangle$ — Replaces the name of a function or predicate by its definition. This command requires that the given $name$ has been explicitly defined, it does not work for predicates or functions defined implicitly through axioms. Expansion can be applied to a single formula, or to the entire proof state.

Example. `expand divides in xon;`

case $\langle formula \rangle$ — Performs case distinction. Given a formula A , this command introduces the tautology $A \vee \neg A$ and applies \vee -elim to it, splitting the proof state into two new states with additional assumptions A and $\neg A$, respectively.

Example. `case x_0=0;`

decompose () — Repeatedly applies \forall -intro and \rightarrow -intro to the current goal, and \exists -elim to all applicable assumptions, introducing new assumptions and Skolem constants. Also splits assumptions using \wedge -elim and \leftrightarrow -elim, applies \rightarrow -elim, and performs simplifications as much as possible.

goal $\langle line \rangle$ — Starts a proof by contradiction. The current goal is negated and added to the assumptions, and replaced by the negation of the formula in the given assumption $line$.

induction $\langle var \rangle$ in $\langle line \rangle$ — Proves the given goal by induction. The goal must be a universally quantified formula, var must be one of the quantified variables, and its type must be \mathbb{N} . If these conditions are satisfied, two new proof states are generated, one for the inductive base and another for the inductive step.


Example. `induction x in cq1;`


instantiate $\langle t_1, \dots, t_n \rangle$ in $\langle line \rangle$ — Applies \forall -elim to an assumption, or \exists -intro to a goal, substituting the given terms for the variables.

Example. `instantiate 0,x_0+1 in bwz;`

lemma $\langle name \rangle$ — Adds a (proved or unproved) formula from the declarations window as a new assumption to the current proof state.

Example. `lemma MULT_1a;`

scatter () — A convenient shorthand for the combination of **decompose** and **split**, introducing Skolem constants and splitting up the proof.

split () — Splits the proof state by applying \wedge -intro and \leftrightarrow -intro to the goal.

split $\langle line \rangle$ — Splits a disjunction or implication in an assumption into subproofs using \vee -elim. Implications $A \rightarrow B$ can also be split by this command, producing (among others) a new subproof with the negated precondition $\neg A$ as a new assumption. You may next want to use the command **goal** $\langle line \rangle$ on this negated precondition formula to achieve the effect of \rightarrow -elim.

For more information please read the tutorial and the manuals.

<http://www.risc.uni-linz.ac.at/research/formal/software/ProofNavigator/>