CS101C Homework 7

Due: Wednesday, May 21, 9PM (firm)

Collaboration: You are allowed and encouraged to work together and collaborate on all aspects of this homework. However, your submission **must** be your own; you must type in your homework without referring to shared or other "external" material. For example, suppose you work as part of a group to prove a long, complicated theorem; and suppose you sketch the proof on the board. When you enter the proof into MetaPRL for homework submission, you must not refer to the board — you must recover the proof from your own memory.

Setup

Start this homework by updating MetaPRL to revision **10** (e.g. version "`0.8.3 (CS101 rev 10)`"). Upgrade instructions are available at `http://nogin.org/cs101c/mp-update.html`.

In this homework you will be extending the `cs101_list2` theory (which is based on portions of the `itt_list2` module) that was presented in the lecture on May 14th. In directory `theories/itt` of your MetaPRL installation, copy

	`cs101_list2.ml`		`cs101_hw7_`*name*`.ml`
the files	`cs101_list2.mli`	into	`cs101_hw7_`*name*`.mli` respectively
	`cs101_list2.prla`		`cs101_hw7_`*name*`.prla`

(where *name* is your login name). In file `cs101_hw7_`*name*`.prla`, replace all occurrences of string `Cs101_list2` with `Cs101_hw7_`*name* (use your favorite text editor's "replace all" functionality). Finally, add `cs101_hw7_`*name* to the end of the `MPFILES` variable in the `theories/int/Makefile`.

For this homework, you should be working within the `ITT` theory. You are not allowed to add any new `prim` rules or rewrites to the system and you are not allowed to modify the system in any way, other than extending it with your new `hw7` module.

Note: after you change the `MPFILES` variable in the `Makefile` or add a new `extends` or `open` directives to a MetaPRL file and before you run `make opt`, you might need to run `make depend` to update the cross-module dependencies.

In all problems of this homework you may add extra well-formedness assumption(s) when it is necessary to make the rules provable. Make sure you only add the one(s) that are truly necessary. Also, you might find it useful

to formulate and prove some "intermediate" lemmas, maybe even define new operator(s).

Part I: Squash and squash–stability

1. Prove the rule

$$\frac{\Gamma;\ \left[B\right] \vdash\ B \qquad \Gamma;\ \left[C\right] \vdash\ C}{\Gamma;\ \left[A \Rightarrow (B \wedge C)\right] \vdash\ A \Rightarrow (B \wedge C)}$$

2. Prove the rule

$$\frac{\Gamma;\ x : \mathrm{Unit};\ \left[A[x]\right] \vdash\ A[x]}{\Gamma;\ \left[\exists x : \mathrm{Unit}.A[x]\right] \vdash\ \exists x : \mathrm{Unit}.A[x]}$$

Part II: Lists

1. Define a `rev` operator that would compute a reverse of a list (e.g. `rev`$\{l\}$ would have the same elements as list l, but in reverse order). Replace the placeholder definition `unfold_rev` with your definition. Optional: if you want to double-check your definition is correct, use it to prove, for example, `rev`$\{a :: b :: c ::$ `nil`$\} \leftrightarrow c :: b :: a ::$ `nil`.

2. State, prove and add to `reduce` resource the reductions of `nil` and `cons` cases of `rev` (similar to those that are already present for `append`).

3. Prove that `rev`$\{$`rev`$\{l\}\}$ is the same as l.

Submission Instructions

Make sure you `export` all the proofs. Send all the files `cs101_hw7_`*name*`.ml`, `cs101_hw7_`*name*`.mli` and `cs101_hw7_`*name*`.prla` as text attachments in an email to `cs101-admin@metaprl.org`. Please include "`CS101 HW7`" in the message subject line.