CS101C Homework 4

Due: Wednesday, Apr 30, 2PM (firm)

Collaboration: For this homework, you can discuss the general principles and ideas as well as the material presented in class, but you should work alone on the assigned problems.

Setup

Start this homework by updating MetaPRL to revision **6** (e.g. version "`0.8.1 (CS101 rev 6)`"). Upgrade instructions are available at `http://nogin.org/cs101c/mp-update.html`.

Next, in directory `theories/cs101` of your MetaPRL installation, create a file `cs101_hw4_`*name*`.ml`, where *name* is your login name (for example, if I was doing this homework, I would create `cs101_hw4_nogin.ml`). Also create the corresponding `.mli` file and add the file name (`cs101_hw4_`*name*) to the `MPFILES` variable in the `theories/cs101/Makefile`.

For this homework, you should be working in the `Cs101_lc` formalization of the λ-calculus. 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 `Cs101_hw4_`*name* 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.

Hint: In many problems of this homework, the automation tactics (such as `autoT` and `cutAssumT`) could help a lot. In fact, some proofs could require only a single `autoT` invocation. Please try using these tactics, when appropriate.

Part I

For each of the following terms, figure out what its type is, add a theorem stating that the term has that type to your `cs101_hw4_`*name* file and prove that theorem.

Note: in case a term has more than one type, you must use the *most general* one. In particular, do not use `Void` and `Unit` constants unless you have to and do not reuse a variable, when using a different variable would also work (e.g. whenever possible, use $A \to B$ instead of $A \to A$, $A \times B$ instead of $A \times A$, etc).

1. $\lambda x.\lambda y.(x,y)$
2. $\lambda x.(\texttt{match}\,x\,\texttt{with}\;\texttt{inl}\{u\} \rightarrow u \mid \texttt{inr}\{v\} \rightarrow \lambda x.(x\,v))$
3. $(\lambda f.(f\,f))\,(\lambda x.(x,x))$

Part II

For each of the following rules, figure out what the ??? needs to be in order for the rule to become *derivable*, add the rule to your `cs101_hw4_`*name* file and prove it. As in Part I, when there are several possibilities for a type, you need to use the most general one.

1.
$$\frac{\Gamma \vdash x \in ??? \qquad \Gamma \vdash y \in ??? \qquad \Gamma \vdash z \in ???}{\Gamma \vdash x\,(y,z) \in T}$$

2.
$$\frac{\Gamma;\, x : A;\, \Delta \vdash t_1[x] \in C \qquad \Gamma;\, x : B;\, \Delta \vdash t_2[x] \in C}{\Gamma;\, x : (A+B);\, \Delta \vdash ??? \in C}$$

3.
$$\frac{\Gamma \vdash x \in (A+B) \qquad \Gamma;\, u : A \vdash t_1[u] \in C \qquad \Gamma;\, v : B \vdash t_2[v] \in C}{\Gamma \vdash (\texttt{match}\,x\,\texttt{with}\;\texttt{inl}\{u\} \rightarrow t_1[u] \mid \texttt{inr}\{v\} \rightarrow t_2[v]) \in C}$$

(this is the `decide_type` rule from `cs101_lc`).

Submission Instructions

First, `export` the proofs to `cs101_hw4_`*name*`.prla` file and submit the `cs101_hw4_`*name*`.ml`, `cs101_hw4_`*name*`.mli` and `cs101_hw4_`*name*`.prla` files. Send the files as text attachments in an email to `cs101-admin@metaprl.org`. Please include "`CS101 HW4`" in the message subject line.

Warning: the `.ml` file you submit **must** compile. Submissions that have syntax errors, or fail to compile for other reasons (for example, failing OCaml type-checker) are likely to only receive partial credit, or **no credit at all**.