Lens Programming By Example

Owen Gunden  
(ogunden@seas.upenn.edu)

William Lovas  
(wlovas@stwing.upenn.edu)

Kate Moore  
(kfm@seas.upenn.edu)

Cyrus Najmabadi  
(cyrus@stwing.upenn.edu)

December 30, 2002
Abstract

This is our abstract!
1 Introduction
end: strong claim!
Whee–harmony is totallysweet. Introduce the general idea of lenses and views.

2 Specification of the Problem

2.1 General Problem
For the purposes of our work, we use the following simplified definitions of lenses and views.
The set of names is denoted by $\mathcal{N}$. It is left unspecified, but it is partitioned a set of category names $\mathcal{C}$
and a set of data names $\mathcal{D}$, such that the following are true:
$$\mathcal{C} \cup \mathcal{D} = \mathcal{N}$$
$$\mathcal{C} \cap \mathcal{D} = \emptyset$$

The set of views is denoted by $\mathcal{V}$. For our purposes, a view is a rooted tree with edges labeled by names.
We represent views as partial functions from names to views:
$$\mathcal{V} = \{ v : (v : \mathcal{N} \to \mathcal{V} \cup \{\bot\}) \}$$
For any $v \in \mathcal{V}$ and $n \in \mathcal{N}$, $v(n)$ is the sub-view of $v$ reached by following an edge labeled $n$, or $\bot$ if there is
no such edge.
The set of lenses is denoted by $\mathcal{L}$. A lens is simply a function from views to views.
$$\mathcal{L} = \{ l : (l : \mathcal{V} \to \mathcal{V}) \}$$
Note that these lenses are uni-directional; there is no notion of a pushdown.

2.2 Specific Problem
$l$ is a metavariable denoting lenses. $N$ is a metavariable denoting subsets of $\mathcal{N}$.
$$l ::= \text{filter } N$$
$$\quad | \text{rename } b \quad b \text{ is a bijection on the set } \mathcal{N}$$
$$\quad | \text{map } l$$
$$\quad | l_1 ; l_2 \quad \text{Sequencing of } l_1 \text{ and } l_2$$
Lens semantics.

3 Solution

3.1 Data Structures
All of our lenses are mostly commutative, so the normal form of a lens simply contains the information about
what was filtered, what was renamed, and what lens, if any, was mapped over the children. Since there is
the possibility that no lens was mapped, we need an explicit identity lens in this representation.
$$l_n ::= \text{id}$$
$$\quad | (N, b, l) \quad \text{representing “filter } N; \text{ rename } b; \text{ map } l”$$

Lens sets are represented similarly as a triple containing all information necessary to infer the set of
lenses referred to. Additionally, a lens set may be empty, or may contain every lens.
\[ L ::= (N_k, N_r, B, L) \]
\[ \mid \Omega \]
\[ \mid \emptyset \]

\(N_k\) is the set of names that must be kept, \(N_r\) is the set of names that must be removed, \(B\) is a representation of a set of bijections on \(\mathcal{N}\), and \(L\) is the set of lenses that may have been mapped.

(What is \(B\)? What could \(B\) be? What does \(B\) end up being?)

3.2 Algorithm

3.3 Correctness of Algorithm

4 Related Work

We din’t do no readin’ sah.

5 Conclusions and Future Work

- integrate with harmony’s concepts and uh - more lenses - do it without restrictions (LDR/SDR) - haskell / meet Santa!