When are two programs equal?

Johan Jeuring

Open Universiteit Nederland and Utrecht University

IFIP WG2.1 Meeting #69, Ottawa, Canada

October 2012

range

Write a function which enumerates all numbers contained in a given range.

$range :: Int \to Int \to [Int]$

For example, *range* 2 5 gives

$[2, 3, 4, 5]$

Some solutions for *range*

$$
\begin{array}{lll}
\mathit{range}_1\ x\ y & = & \textbf{if}\ x \equiv y\ \textbf{then}\ [x]\ \textbf{else}\ x : \mathit{range}_1\ (x+1)\ y \\
\mathit{range}_2\ x\ y & = & \textbf{if}\ y \equiv x\ \textbf{then}\ [x]\ \textbf{else}\ x : \mathit{range}_2\ (x+1)\ y \\
\mathit{range}_3\ x\ y & = & \textbf{if}\ x \not\equiv y\ \textbf{then}\ x : \mathit{range}_3\ (x+1)\ y\ \textbf{else}\ [x] \\
\mathit{range}_4\ x\ y & = & \textbf{if}\ y \not\equiv x\ \textbf{then}\ x : \mathit{range}_4\ (x+1)\ y\ \textbf{else}\ [x] \\
\mathit{range}_5\ x\ y & = & \textbf{if}\ x \not\equiv y\ \textbf{then}\ x : \mathit{range}_5\ (1+x)\ y\ \textbf{else}\ [x] \\
\quad \text{-- and the 3 variants} & & \\
\mathit{range}_6\ x & = & \lambda y \rightarrow \textbf{if}\ x \equiv y\ \textbf{then}\ [x]\ \textbf{else}\ x : \mathit{range}_6\ (x+1)\ y \\
\quad \text{-- and the 7 variants} & & \\
\mathit{range}_7 & = & \lambda x \rightarrow \lambda y \rightarrow \textbf{if}\ x \equiv y \\
 & & \qquad\qquad\qquad \textbf{then}\ [x] \\
 & & \qquad\qquad\qquad \textbf{else}\ x : \mathit{range}_7\ (x+1)\ y \\
\quad \text{-- and the 7 variants} & &
\end{array}
$$

A procedure for determining equality

- A procedure for determining whether or not two programs are equal is necessarily going to have some limitations
- But surely each pair of *range* programs can pass the test
- How can determine many of these equalities?
- What program transformations can I specify to steer this procedure?

I need a normal form!

- Remove syntactic sugar
- Normalization by Evaluation normalizes based on types, so a function of type $a \rightarrow b \rightarrow c$ always has the form $\lambda x \rightarrow \lambda y \rightarrow \ldots$
- Normal forms for integer expressions, boolean expressions, string expressions, taking into account algebraic properties of the operators
- Inlining?

> $\mathbf{let}\ \mathit{duplicate}\ x = [x, x]\ \mathbf{in}\ \mathit{concatMap}\ \mathit{duplicate}$
> $\mathit{concatMap}\ (\lambda x \rightarrow [x, x])$

- Fusion? $\mathit{map}\ f \circ \mathit{map}\ g = \mathit{map}\ (f \circ g)$

Problems

- High-level: how can I determine equality of (functional) programs?
- What is a normal form of a program?
- What sequence of steps do I use for determining a normal form of a program?
- How can I influence the computation of a normal form of a program?