**Translation Correctness for Expressions**

Recall [Evaluating Postfix](http://lara.epfl.ch/web2010/cc09:evaluating_postfix)

**Notation**

Functions:

* evaluation: $eval\ :\ Expression \to Int$
* compilation: $[\ \_ ]\!]\ :\ Expression \to List[Instruction]$
* virtual machine: $run\ : \ List[Instruction] \to List[Int] \to List[Int]$

For simplicity we will omit environments mapping variables

* does not affect this proof

We often write X instead of List(X) when X is one element

Also, we often use same symbol ‘\*’ for

* operator in the tree
* operator in instruction sequence
* corresponding mathematical operator

**Key Facts**

**Evaluation of expressions:**   
![\begin{displaymath}
   eval(e_1 * e_2) = eval(e_1) * eval(e_2)
\end{displaymath}

**Translation:**   
\begin{displaymath}
   [\![ e_1 * e_2 ]\!] = [\![e_1]\!] ::: [\![e_2]\!] :::\ {*}
\end{displaymath}  
Here

* the last occurrence of is the imul instruction
* $e_1*e_2$denotes a tree with node type $*$and children $e_1$, $e_2$

**run-virtual machine**: From definition of $run$and by induction on $L_1$,   
\begin{displaymath}
    run\ (L_1 ::: L_2)\ S = run\ L_2\ (run\ L_1\ S)
\end{displaymath}  
Of course   
\begin{displaymath}
    run\ (* ::: L)\ (S:::x_1:::x_2) = run\ L\ (S:::x)
\end{displaymath}  
where $x$is the result of computing $x_1*x_2$, and   
\begin{displaymath}
    run\ (c ::: L)\ S = run\ L\ (S:::c)
\end{displaymath}  
for constant $c$, and   
\begin{displaymath}
    run\ EmptyList\ S = S
\end{displaymath}

**Correctness Theorem**

For every $e$,   
\begin{displaymath}
   run\ [\![ e ]\!]\ List()\ \ =\ \ List(eval(e))
\end{displaymath}

**Attempt at Inductive Proof**

Induction on structure of $e$

\begin{displaymath}\begin{array}{l}
   run\ [\![e_1 * e_2]\!]\ S = \\
   run\ ([\![e_1]\!] ::: [\![e_2]\!] ::: *)\ S = \\
   run\ ([\![e_2]\!] ::: *)\ (run\ [\![e_1]\!]\ S) = \\
   ?
\end{array}\end{displaymath}

**Induction Hypothesis**

For every expression $e$and every list of integers $S$,   
\begin{displaymath}
   run\ [\![ e ]\!]\ S\ \ =\ \ S ::: eval(e)
\end{displaymath}

**Inductive Proof**

\begin{displaymath}\begin{array}{l}
   run\ [\![e_1 * e_2]\!]\ S = \\
   run\ ([\![e_1]\!] ::: [\![e_2]\!] ::: *)\ S = \\
   run\ ([\![e_2]\!] ::: *)\ (run\ [\![e_1]\!]\ S) = \\
   run\ ([\![e_2]\!] ::: *)\ (S ::: eval(e_1)) = \\
   run\ (*)\ (run\ [\![e_2]\!]\ (S ::: eval(e_1))) = \\
   run\ (*)\ (S ::: eval(e_1) ::: eval(e_2)) = \\
   S ::: (eval(e_1) * eval(e_2)) = \\
   S ::: eval(e_1 * e_2)
\end{array}\end{displaymath}