Day 4

1. Evaluation May Be Partial

Let's extend our language a little:

\[ T \ni t := z \mid t_1 + t_2 \mid t_1 \times t_2 \mid t_1 - t_2 \mid t_1 \div t_2 \]

and correspondingly, extend our evaluation relation:

\[
\begin{align*}
\vdash t_1 \Downarrow z_1 & \quad t_2 \Downarrow z_2 & \quad t_1 \Downarrow z_1 & \quad t_2 \Downarrow z_2 \\
& (z_2 \neq 0)
\end{align*}
\]

Is our evaluation relation still total? Deterministic?

2. Characterizing Partiality

We could attempt to characterize when our evaluation relation does hold. We'll begin by extending the ± semantics to incorporate the new cases. Again, we need some lookup tables.

Using them, we can define new inference rules for evaluation (or evaluation rules)

\[
\begin{align*}
\vdash t_1 \Downarrow S_1 & \quad t_2 \Downarrow S_2 \\
& (t_1 - t_2) \Downarrow \bigcup \{s_1 - s_2 \mid s_1 \in S_1, s_2 \in S_2\}
\end{align*}
\]

We already have some neat results. Consider:

\[
\begin{align*}
6 \Downarrow \{+\} & \quad 0 \Downarrow \{0\} \\
& 6 \div 0 \Downarrow \emptyset
\end{align*}
\]

but unfortunately:

\[
\begin{align*}
6 \Downarrow \{+\} & \quad 6 \Downarrow \{+\} \\
& 6 \div (6 - 6) \Downarrow \{-, +\}
\end{align*}
\]
3. The Goal

Key idea: $\Downarrow_\pm$ over-approximates the behavior of $\Downarrow$. So while we have a guarantee one direction:

$$t \Downarrow z \implies t \Downarrow_\pm S \land \text{signum}(z) \in S$$

we do not have a guarantee the other direction:

$$t \Downarrow_\pm S \land s \in S \not\implies t \Downarrow z \land \text{signum}(z) = s$$

given

$$\text{signum}(z) = \begin{cases} - & \text{if } z < 0 \\ 0 & \text{if } z = 0 \\ + & \text{otherwise} \end{cases}$$

So $\Downarrow_\pm$ doesn’t fully characterize when terms evaluate, although it gets us much of the way there. To get the rest of the way, I’ll define a relation called “safety”:

<table>
<thead>
<tr>
<th>$n$ safe</th>
<th>$t_1$ safe</th>
<th>$t_2$ safe</th>
<th>$t_1$ safe</th>
<th>$t_2$ safe</th>
<th>$t_1$ safe</th>
<th>$t_2$ safe</th>
<th>$t_2 \Downarrow_\pm S$</th>
<th>0 $\not\in$ S</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1 + t_2$ safe</td>
<td>$t_1 - t_2$ safe</td>
<td>$t_1 \times t_2$ safe</td>
<td>$t_1 \div t_2$ safe</td>
<td>$t_1 \Downarrow_\pm S$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Why not just $t_1 + t_2$ safe? Addition never “goes wrong”.

Is safety sufficient (sound)? Want: $t$ safe $\implies \exists z.t \Downarrow z$.

Is safety necessary (complete)? Want: $\exists z.t \Downarrow z \implies t$ safe.

Summary: Just as $\Downarrow_\pm$ over-approximates the behavior of $\Downarrow$, safety under-approximates the behavior of $\Downarrow$.

3. The Goal

In general, can we have a sound and complete characterization of a property like safety?

No! Rice’s theorem says that any non-trivial property of the partial computable functions is itself undecidable.

(Reduction to halting problem: given program $p$ input $x$, is the function $y \mapsto p(x)$; $y$ the identity function?)

But does this mean that we can’t prove anything about programs? No! We certainly can prove that $y \mapsto y$ is the identity function.

The goal: identify useful subsets of programs for which desirable properties are provable.