Homework # 2
due March 4, 13:00

The natural language explanations (§2) and research surveys (§4) should be turned in on paper at the *beginning* of lecture on Tuesday, March 4. The SASyLF proof (§3) should be submitted by email to `scmalte@inf.ethz.ch` *before* 1pm on Tuesday, March 4.

1 Reading

Please read through the end of Chapter 3 in your textbook.

2 Problems

Please do problem 3.5.13 (Funny rules). Explain your answers, but no proof (natural language or SASyLF) is required. Also answer the following question:

> Which of the theorems 3.5.4, 3.5.7, 3.5.8, 3.5.11, 3.5.12 remain true after adding the rules for arithmetic expressions (Figure 3–2)? Explain!

(Ex. 3.5.14 and its solution should help, as should definition 3.5.15.)

Theorem *Summary*	3.5.4 determ	3.5.7 valnorm	3.5.8 normval	3.5.11 unique	3.5.12 termination
Add E-FUNNY1	false	true	true	false	true
Add E-FUNNY2	false	true	true	true	true
Add Numbers	true	true	false	true	true

Explanation

Adding E-FUNNY1 drastically changes semantics:

```
if true then false else true
```

can go either to `false` or to `true` in one step which are counter-examples to determinancy (3.5.4) and uniqueness of normal forms (3.5.11).

Adding `E-Funny2` allows us to pre-evaluate the "then" part which means we don't have determinancy (3.5.4) but it doesn't fundamentally change the results of the computation (3.5.11).

Adding numbers adds a lot of new "normal forms" such as `succ true`, which are not values and hence counter-examples to all normal forms being values (3.5.8) but the definitions are carefully crafted to retain determinancy (3.5.4) and hence uniqueness of normal forms (3.5.11).

None of the additions cause values to be evaluable (3.5.7), and none of them can lead to loops in evaluation (3.5.12).

3 Proofs

Do problem 3.5.17 and write the proof in SASyLF (**only** for the "if" sublanguage!). More precisely, prove that if $t \xrightarrow{*} t'$ and t' is a value then $t \Downarrow t'$, and conversely if $t \Downarrow t'$, then $t \xrightarrow{*} t'$. (I am not requiring you to prove that t' is a value.) You may use the solution in the back of the book (p. 498 in my edition) to help you write the proof.

When I solved this problem, I noticed I needed the following lemmas:

1. If $t \to t'$ and $t' \Downarrow v$, then $t \Downarrow v$.
2. If $t \xrightarrow{*} t'$ and $t' \Downarrow v$, then $t \Downarrow v$.
3. If $t_1 \xrightarrow{*} t_1'$ then for any t_2 and t_3, $\texttt{if } t_1 \texttt{ then } t_2 \texttt{ else } t_3 \xrightarrow{*} \texttt{if } t_1' \texttt{ then } t_2 \texttt{ else } t_3$.

Adding `succ`, `pred`, `iszero` doubles the size of the proof—be thankful you don't need to handle them! We will install a "skeleton" file on the web page to get you started.

4 Application

Find three papers from journals or academic conferences in programming languages (TOPLAS, POPL, OOPSLA, ECOOP) from the last 5 years which define an operational semantics (evaluation) for a programming language. For each one, determine whether the evaluation relation is

`small-step` as with the book (with errors getting the program "stuck");

`small-step with errors` as in Exercise 3.5.16

`big-step` as in Exercise 3.5.17

Cite each paper and explain your categorization.