Final report, Jackie Liu, jl2627

Vision

The preliminaries section of the demo provide a good executive summary. Although, the vision has changed somewhat. I think more of the K definitions as a "reference implementation" that is amendable to formal analysis and other language development, rather than executable formal semantics. The theoretical foundation of the framework and its logic still elude me.

Summary of progress

The global environment and lambda abstractions were switched to include a stack-based local one with closures, allowing for a more true syntax and semantics rather than hacks. Object-orientation was introduced to model how contracts exist in state. Arbitrary transactions instead of just one call, in JSON-esque format. Global functions that resemble VM opcodes. Fully working semantics with non-trivial programs introduced in demo.

Activity breakdown

n/a

Productivity analysis

There were much less frequent head-scratching, stuck moments this split that slowed productivity. I accomplished a lot of what I had planned, but did not quite get to analyze more than just the programs in the demo. The K-framework documentation is quite scattered and further analysis would require another big investment in learning some more tools. In this regard I did not get to my excellent scope items such as using the model checker. However, I did get the semantics working in a good state, something which I doubted I could accompolish at many points. All in all, I underestimated the difficulty in opting to use K, but I am happy with what I got done.

Grade

GOOD. This split the final product has changed considerably and is much more useable and full-featured compared to earlier versions. I feel what I accomplished was high enough in complexity. Learning to use K was very intricate and the language developed is non-trivial. What was attempted was outside of familiar territory, perhaps too much so :).

Next steps

The definitions are quite brittle and require a lot of checks on behavior that should not occur, like executing transactions within contracts for example. Type checking would help in this regard as well as a more strict grammar. Not all functionality is

fully implemented, such as atomic transactions. Fixing these rough edges and completing the full functionality would increase the robustness of the definitions. There are also many more K tools in the ecosytem to explore, such as the program analysis tool RV-Match. After these things are done and my proficiency increases, I would expand the semantics beyond the current minimal features to be more useful for a variety of programs as I have mentioned before.