0.0.1 Auto-parallelizing Pure Functional Language System

Report by:	Kei Davis
Participants:	Dean Prichard, David Ringo, Loren Anderson, Jacob Marks
Status:	active

The main project goal is the demonstration of a light-weight, higher-order, polymorphic, pure functional language implementation with which we can experiment with automatic parallelization strategies, varying degrees of default function and constructor strictness, and lightweight instrumentation.

We do not consider speculative or eager evaluation, but do plan to infer strictness by program analysis, so potential parallelism is dictated by the specified degree of default strictness, language constructs for parallelism, and program analysis.

Our approach is similar to that of the Intel Labs Haskell Research Compiler: we use GHC as a front-end to generate STG, then exit to our own back-end compiler. As in their case we do not attempt to use the GHC runtime. Our implementation is *light-weight* in that we are not attempting to support or recreate the vast functionality of GHC and its runtime. This approach is also similar to Don Stewart's except that we generate C instead of Java.

Current Status

Currently we have a fully functioning serial implementation and a primitive proof-of-design parallel implementation. The most recent major development was the bridge between GHC and our system. Thus we can now compile and run Haskell programs with simple primitive and algebraic data types.

Additionally, we have developed a new strictness analysis technique that will be incorporated into the compiler. We'll miss the Static Analysis Symposium 2017 submission deadline, unfortunately.

Immediate Plans

We are currently developing a more realistic parallel runtime. The implementation of the strictness analysis is scheduled as a Master's thesis project starting May 2017.

Undergraduate/post-graduate Internships

If you are a United States citizen or permanent resident alien studying computer science or mathematics at the undergraduate level, or are a recent graduate, with strong interests in Haskell programming, compiler/runtime development, and pursuing a spring, fall, or summer internship at Los Alamos National Laboratory, this could be for you.

We don't expect applicants to necessarily already be highly accomplished Haskell programmers—such an internship is expected to be a combination of further developing your programming/Haskell skills and putting them to good use. If you're already a strong C hacker we could use that too.

The application process requires a bit of work so don't leave enquiries until the last day/month. Dates for terms beyond fall 2017 are best guesses based on prior years.

Term	Application Opening	Deadline
Fall 2017	Open	May 31 2017
Spring 2018	July 2017	Oct 2017
Summer 2018	Oct 2017	Jan 2018

Email kei (at) lanl (dot) gov if interested in more information, and feel free to pass this along.

Further reading

Email same address as above for the Trends in Functional Programming 2016 paper about this project.

Intern Loren Anderson did an interesting Haskell exercise while here for this mathematics paper.