Using Linear Types to bring Functional Programming to Embedded Systems

Michael Nolan

April 18, 2017

Contents

1 Abstract

Abstract here

2 Purpose

To develop a compiler that incorporates linear types in its compilation strategy to reduce the need for a garbage collector.

3 Summary

Statically typed functional programming languages offer better safety guarantees and reduce programmer errors better than almost all other programming languages. These languages, previously relegated to use by academics for PhD theses, are now being used by businesses because of their greater reliability and rapid development. Unfortunately, functional programming languages generate a large amount of garbage and do not allow the programmer to allocate memory on their own. This means that they are unsuitable for use in embedded systems, where realtime guarantees must be made and there is little memory availible for use.

However, the addition of linear types[1] to the language allows the programmer to tell the compiler that certain variables will be used *exactly once*. This means that the compiler can free the variable as it is used and be sure that memory will not be leaked or freed twice. Additionally, the compiler can fuse computations[2, 3], meaning that intermediate values do not need to be constructed. Finally, linear types provide additional safety when used properly, guaranteeing that an action happens exactly once, which is useful in a client-server architecture.

4 Methodology

5 References

[1] J.-Y. Girard, "Linear logic," *Theoretical Computer Science*, no. 50, pp. 1–102, 1988.

[2] "Integration of linear types into ghc."

[3] J.-P. Bernardy and A. Spiwack, "Linear types make performance more predictable," March 2017.