Evolution of the Spineless Tagless G-Machine

Armin Bernstetter

Julius Maximilian University, Würzburg

Abstract. The spineless tagless G-machine (STGM) is an abstract machine that is located at the core of the Glasgow Haskell Compiler GHC. Since its creation at the start of Haskell development in early 1990s it has undergone several significant changes. This work aims at showing the evolution of the STGM and overall at providing insight in the workings of the most used Haskell compiler GHC. [2]

1 Introduction

2 Basics

2.1 Haskell

What is Haskell, where did it start what does it do? [1]

2.2 Compilers

How do compilers work in general

2.3 Abstract machines

What is an Abstract machine?

3 GHC

The Glasgow Haskell Compiler, named after the scottish city

3.1 Core Language

3.2 The STGM Language

3.3 $C--$

3.4 Backends

C Backend

Native Code Generator

LLVM

4 STGM in depth

Spineless

Tagless

4.1 Evaluation

Push/Enter

Eval/Apply

4.2 Dynamic Pointer Tagging

[3]

5 Conclusion

6 First Section

6.1 A Subsection Sample

Please note that the first paragraph of a section or subsection is not indented. The first paragraph that follows a table, figure, equation etc. does not need an indent, either.

Subsequent paragraphs, however, are indented.

Sample Heading (Third Level) Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

Sample Heading (Fourth Level) The contribution should contain no more than four levels of headings. Table 1 gives a summary of all heading levels.
Displayed equations are centered and set on a separate line.

$$x + y = z \tag{1}$$

Please try to avoid rasterized images for line-art diagrams and schemas. Whenever possible, use vector graphics instead (see Fig. 1).

Theorem 1. *This is a sample theorem. The run-in heading is set in bold, while the following text appears in italics. Definitions, lemmas, propositions, and corollaries are styled the same way.*

Table 1. Table captions should be placed above the tables.

Heading level	Example	Font size and style
Title (centered)	**Lecture Notes**	14 point, bold
1st-level heading	**1 Introduction**	12 point, bold
2nd-level heading	**2.1 Printing Area**	10 point, bold
3rd-level heading	**Run-in Heading in Bold.** Text follows	10 point, bold
4th-level heading	*Lowest Level Heading.* Text follows	10 point, italic

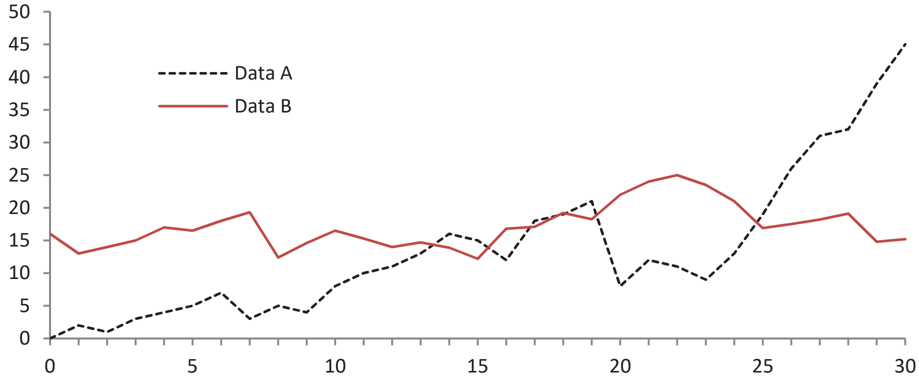

Fig. 1. A figure caption is always placed below the illustration. Please note that short captions are centered, while long ones are justified by the macro package automatically.

Proof. Proofs, examples, and remarks have the initial word in italics, while the following text appears in normal font.

For citations of references, we prefer the use of square brackets and consecutive numbers. Citations using labels or the author/year convention are also acceptable. The following bibliography provides a sample reference list with entries for journal articles [**?**], an LNCS chapter [**?**], a book [**?**], proceedings without editors [**?**], and a homepage [**?**]. Multiple citations are grouped [**?**,**?**,**?**], [**?**,**?**,**?**,**?**].

References

1. Hudak, P., Hughes, J., Peyton Jones, S., Wadler, P.: A history of haskell: being lazy with class. In: Proceedings of the third ACM SIGPLAN conference on History of programming languages. pp. 12–1. ACM (2007)
2. Marlow, S., Jones, S.P.: Making a fast curry: push/enter vs. eval/apply for higher-order languages. In: ACM SIGPLAN Notices. vol. 39, pp. 4–15. ACM (2004)
3. Marlow, S., Yakushev, A.R., Peyton Jones, S.: Faster laziness using dynamic pointer tagging. In: Acm sigplan notices. vol. 42, pp. 277–288. ACM (2007)