Temporary Sucky title

Christian Chavez

Nico Reissmann, Magnus Jahre, and Christian Chavez are with the Norwegian University of Science and Technology (NTNU).

***Index Terms*—Function inlining, Jive, Compiler, 2015, NTNU**

***Abstract*—Lorem ipsum...**

I. Dictionary

Figure out the terms found in papers...

A. placeholder

Insert reference to “GHC secrets” -paper in subsection title above

- Section 1
  - “inlining subsumes”
  - “lexical scopes”
  - “pure” (*language*)
  - “explicitly typed” (*language*)
  - “strictness analysis”
  - “*let*-floating” (*Haskell*)
  - “name capture”
- Section 2
  - “β-reduction”
  - “invariant” (*language artifact/variable/expression?*)
  - “*trivial-constructor-argument invariant*” (*Haskell?*)
  - “divergent computations”
  - “closure” (*scopes of functions?*)
  - “lambda calculus”
  - “literals”
  - “primitive operators”
- Section 3
  - “bound variable”
  - “recursive binding groups”
  - “strongly-connected components”
  - “one-shot lambdas”
  - “contravariantly” (*(..) it appears contravariantly in its own definition.*)
  - “ùntyped programs”
  - “pathological programs”
  - “static analysis”
- Section 4
  - “hash-consing”

II. Introduction

III. Background

In this section we first explain why inlining is a practice found in almost every compiler to date, before we list (in order of relevance)the papers contributing to the background of this report, and finally summarize what each paper brings to this report.

To do

A. Inlining

Decide whether to keep or throw out.
If throw out, perhaps re-arrange structure wrt. Related Work.

B. Related Work

- “**Secrets of the Glasgow Haskell Compiler inliner**”, written by Simon Peyton Jones and Simon Marlow, at Microsoft.

  Summary of paper.

- “**Automatic Autotuning of Inlining Heuristics**”, written by John Cavazos and Michael F.P. O’Boyle, at School of Informatics University of Edinburg.

  Summary of paper.

IV. Scheme

V. Methodology

VI. Results

VII. Discussion

VIII. Conclusion

A. Further Work