Creating a Compiler Complete with Type Checker, as Research into a Core Language for Type Analysis

Thomas Butterwith 090004683 **Supervisor:** Marco Gaboardi

Aims
The aim of this project is to create a compiler, capable of implementing arithmetic as well as lambda calculus, complete with type checking. Time permitting, the project will also allow for comparison with the compiler described in *Type-based Sensitivity Checking* (L D'Antoni et al).

Prototypes
As part of the project, a simple compiler to handle arithmetic calculations was written and debugged towards the end of 2014. Leading up the creation of the basic compiler, research and practical examples were undertaken in OCaml, to create a foundation of knowledge before attempting the project. This included a reading and basic understanding of the *OCaml Reference Manual* (Leroy 2013), with *Real World OCaml* (Minsky et al 2014) providing a number of code examples and tutorials to follow.

Writing a basic compiler created a starting block to build the rest of the project around, starting with the addition of lambda calculus followed by type checking.

Literature
Along with the OCaml Reference Manual, a number of texts have been useful in the progression of the project. Real World OCaml was used to gain an understanding of OCaml as well as simple compiler writing. To provide a basic understanding of lambda calculus, *A Tutorial Introduction to the Lambda Calculus* by Raul Rojas and *Introduction to Lambda Calculus* by Henk Barendregt and Erik Barendsen were both utilised and studied.

Methodology
The project is easily separable into smaller chucks of functionality, each one requiring reading and research, followed by implementation and finally testing. By separating the project into these pieces and allocating a set amount of time to each stage of each piece allows the project to be carefully controlled. As the project is a research based project into compilers, no user testing is needed saving time overall that can be utilised for further development of the compiler.

Progress
Over the next few weeks, much of the time available will be spent researching and learning about lambda calculus and compiler design, including small programatic tests in OCaml. Once a level of confidence has been reached lambda calculus will be added to the original arithmetic compiler.

Once the lambda calculus functionality has been added to the compiler, a week has been put aside to fully test the compiler and complete the write up of notes about the process.

Beyond lambda calculus the next step is to implement type checking. Five weeks have been set aside for research into type checking and existing type checking implementations in other compilers. This is an ample amount of time to read around the subject and gain a satisfactory understanding of the subject. Like lambda calculus, once an understanding has been gained five more weeks have been set aside to implement type checking with a further week to test the functionality.

At this point, if the project is on schedule, there should be two weeks to compare this project with the *Type-based Sensitivity Checking* project mentioned previously.

Towards the end of the project, three weeks have been set aside to compile all notes taken throughout the project into a first draft of the report with a further week to redraft the report for submission.

Difficulties
Foreseen difficulties include a difficulty in understanding and implementation of lambda calculus at an early stage of the project. This can be overcome with regular conversation with the supervisor via skype. Similarly, any lack of understanding or technical problems with type checking could pose a threat to the project's progression. Again, through regular conversation with the supervisor and sufficient reading around the subject should negate this problem.

Potential technical errors include an inability to compile properly. This problem arose early on in the project when attempting to compile the arithmetic compiler and a lack of documentation online made the issue difficult to resolve.

Title	Duration	Start	End
Test Arithmetics	1w 1d	30/01/2015 08:00	06/02/2015 17:00
Reseach Lambda Calculus	1w 1d	30/01/2015 08:00	06/02/2015 17:00
Research Compiler Design	1w 1d	30/01/2015 08:00	06/02/2015 17:00
Implement Lambda calculus into compiler	4w	09/02/2015 08:00	06/03/2015 17:00
Test Lambda Calculus	1w	09/03/2015 08:00	13/03/2015 17:00
Research Type Checking	5w	09/02/2015 08:00	13/03/2015 17:00
Implement type checking	5w	16/03/2015 08:00	17/04/2015 17:00
Test Type Checking	1w	20/04/2015 08:00	24/04/2015 17:00
Write Report First Draft	3w	30/03/2015 08:00	17/04/2015 17:00
Redraft Report	1w	20/04/2015 08:00	24/04/2015 17:00
Create Presentation	1w	27/04/2015 08:00	01/05/2015 17:00