Shi Shu

Fitzwilliam College

ss2099

Computer Science Part II Project Proposal

An Exact Real Package for ML

24th October 2013

Project Originator: Dr. Arno Pauly

Supervisors: Dr. Arno Pauly

Director of Studies: Dr. Robert Harle

Overseers: Dr. Richard Gibbens / Prof. Larry Paulson

1.Introduction

Many applications of computers require the use of real arithmetic. Unfortunately, there are a number of significant problems associated with performing real arithmetic using floating point approximation, the method most commonly used by programmers, and which forms part of most computer languages and the instruction set of many modern CPUs. These problems stem from the fact that only a finite set of reals are represented exactly using this approach, and rounding occurs after each floating point operation. They can result in complete loss of accuracy in even relatively simple computations.

Originally, the idea of the project came from a paper "A new introduction to the theory of represented space" by Dr. Arno Pauly (See *http://arxiv.org/pdf/1204.3763v2.pdf*). Following from the paper, a synthetic and functional approach to solve differential equation can be derived. Hence implementing the algorithm in a functional language would be straight forward. Also the algorithm would require the use of exact real numbers, continuous functions, open sets and so on. Thus ML can be a good choice of programming language here, which then directly lead to core part of the project, developing an exact real package for ML.

There are many existing exact real packages in different languages using different implementations. Among them, the iRRAM (See *http://irram.uni-trier.de/*) is a very efficient C++ package for error-free real arithmetic based on the concept of a Real-RAM. Its capabilities range from ordinary arithmetic over trigonometric functions to linear algebra even with sparse matrices.

The project mainly focuses on building an exact real package, evaluating its performance and comparing it with other exact real implementations (iRRAM in C++). Then completing the algorithm of the differential equation solver using the package will be an extension to the project. As a simpler example, it can be proved that the inverse of a continuous injection from a compact and admissible space to a Hausdorff space can be computed. Hence implementing an inverse function calculator using the package can be another extension.

In terms of representing real number, "Computable Analysis" by Professor Klaus Weirauch discusses the problem from a theoretical prospective in detail and can be used as a starting point. And also classical complexity theory would not work with infinite lists (and subsequently, real numbers etc). "Complexity theory for operators in analysis" by Akitoshi Kawamura and Stephen Cook (See *https://www.cs.toronto.edu/~sacook/homepage/stoc.2010.pdf*) describes how complexity theory can be applied to this setting.

2. Resource Required

The core part of the project will be written in Standard ML.

The iRRAM package of C++ will be needed to compare with the ML implementation.

Both my own laptop and desktop in Intel Lab will be used for programming.

Backup plans will include Dropbox and Google Drive.

GitHub will be used for version control.

LaTex will be used for dissertation writing.

3. Starting Point

The project will be undertaken with the background knowledge acquired from the following courses in previous years :

Programming in ML from Part IA course.

Computation Theory from Part IB course.

4. The Substance and Structure of the Project

The objective of the project is to build an exact real arithmetic package for ML and compare the implemented package with the other existing exact arithmetic packages available for other languages(i.e. iRRAM). A competition between existing exact real systems is conducted by Jens Blanck (See *http://www-compsci.swan.ac.uk/~csjens/pdf/20640389.pdf*). A similar testing method might be used here.

Thus the project contains 2 main parts and 2 extensions.

1. Implement an exact real arithmetic package for ML.
   A. Find a suitable representation of real numbers in ML.
   B. Develop algorithms for all operations under such representation.
   C. Implement such data structures and operations in ML.
   D. Compute the complexity of all the operations both in terms of time and memory for this implementation.
2. Compare the implemented package with other packages in terms of time and memory efficiency.
3. (Extension) Implement an inverse function calculator using the package.
4. (Extension) An algorithm of solving differential equations can be derived from the theory of represented space. Implement the algorithm using the exact real packages built in ML to construct a DE solver.

5. Success Criteria

For the project to be considered successful the following items must be completed.

1. A working exact real package in ML should be delivered.

2. The performance of the implementation should be evaluated.

Further to the core part of the project, the extension can be regarded as successful if a working inverse function calculator is implemented and the differential equation solver is able to solve DE correctly.

6. Timetable and Milestones

The timetable below is divided into 2-4 week slots.

Week 1 and 2 (Oct. 24th to Nov. 6th)

1. Research possible ways of representing exact reals. Read related material in "Computable Analysis".

2. Understand some of the current implementations of exact real arithmetic in other languages.

3. Learn LaTex to prepare for dissertation writing.

Week 3, 4, 5 and 6 (Nov. 7th to Dec. 4th)

1. Find the suitable representation of real numbers in ML.

2. Implement the real numbers representation in ML.

Week 7, 8 and 9 (Dec. 5th to Dec. 25th)

1. List all the operations that the real number should be able to support.

2. Design algorithms for every operation.

Week 10 and 11 (Dec. 26th to Jan. 8th)

1. Implement all the exact real operations.

2. Create test data to test the package.

Week 12, 13 and 14 (Jan. 9th to Jan. 29th)

1. Debug the system.

2. Evaluate the complexity of all the operations under such implementation.

3. Write the progress report.

4. Prepare the presentation.

Week 15 and 16 (Jan. 30th to Feb. 12th)

1. Start writing the implementation part of the dissertation.
2. Test the implemented package against iRRAM.
3. Analyse the reasoning behind the possible performance difference.

Week 17 and 18 (Feb.13th to Feb. 26th)

1. Implement the inverse function solver using the package.
2. Start writing the evaluation part of the dissertation.

Week 19, 20 and 21 (Feb. 27th to Mar. 19th)

1. Implement the abstract algorithm of the DE solver in ML.
2. Add Extension to the draft dissertation.

Week 22, 23 and 24 (Mar. 20th to Apr. 9th)

1. Finish the draft dissertation.
2. Make the implemented package support exception handling.

Week 25 and 26 (Apr. 10th to Apr. 23rd)

1. Polish dissertation and formatting dissertation correctly.
2. Add reference and code to dissertation.

Week 28 and 29 (Apr. 24th to May 7th)

1. Kept in case any of the previous steps takes longer than expected.

Week 30 (May 8th to May 15th)

1. Finalise the dissertation.
2. Print and hand in the dissertation.