PhD Project-Plan

Jonathan THALER

March 6, 2017

This document gives detailed information about the structuring of the research undertaken in this PhD.

1 Years

The whole PhD lasts for 3 years, 36 Months, from October 2016 to September 2019 and thus I will structure it according to 3 years where each year will be a major milestone - which is also intended by the Computer School.

1.1 1st Year: Groundwork

In this year I will learn basics and develop and research the methodology I will use for the main work in the 2nd year. Also I want to write a paper about this work and try to publish it on a Conference or Journal. The time-frame will be set by the date of the 1st year annual oral report which will happen beginning of July thus there are about 6 Months time (counting including January '17 and leaving out July '17). These are the things I want / need to achieve this year:

- Prototyping in Haskell, Scala and Java
- Study Actor-Model theory: Hewitt, Greif, Clinger, Agha
- Get into reasoning about programs
- Basics of Economics [?], [?]
- Basics of ACE: Tesfatsion
- Implement PureAgents Library
- Write & publish papers
- Write 1st year report
- Prototype ACE using Akka and Haskell

Important and mile-stones:

End of March	Finished and submit Paper
June (Mid)	Finished writing 1st year report
July	Oral annual report

1.1.1 January

Research Finish implementation work in Java and Haskell

Paper First draft

Reading

- Book "Gleichzeitige Ungleichzeitigkeiten"
- Complex Systems Stuff
- Find semantics of simulation papers

1.1.2 February

Research

- Theoretical: Specification language for ABS in Haskell.
- Programming: ABS and Yampa.

Paper Refine paper: read by supervisors and colleagues

Reading

- Re-Read existing papers on functional programming and simulations
- Read Semantics of Actors (Greif & Clinger)
- ACE papers

1.1.3 March

Research

- Theoretical: Specification language for ABS in Haskell.
- Programming: ABS and Yampa.

Reading

- Read Semantics of Actors (Greif & Clinger)
- ACE papers

Paper Finalize and submit

1.1.4 April

Research ABS and Yampa

Paper Write potential conference papers on "specification language" and "reactive agents"

Reading ACE papers

1.1.5 May

Paper Continue writing potential conference papers on "specification language" and "reactive agents"

Reading Read papers not read so far but include in 1st year report

Writing Start writing 1st year report

1.1.6 June

Research Do 1st Year oral exam

Reading Read papers not read so far but include in 1st year report

Writing Finalize 1st year report

1.1.7 July to September

Research Prototyping ACE using my Haskell-Library and Akka: bilateral trading, EDSL for ACE and qualitative modelling, reasoning in ACE

Reading

- ACE basics papers (Tesfatsion)
- ACE models for trading/bartering (e.g. Gintis)

1.1.8 October

2nd year starts

1.2 2nd Year: Main Work

Applying 1st year results, methods and experiences to develop and write main paper to be published in a journal in 3rd year thus in 2nd year the main work and implementation will be done. The idea is to start from Ionescus Framework [**?**] and build on his paper.

- Implement Ionescous framework using the methodology developed in 1st year

- Generalize implementation to market models

- Learn Agda and dependent types

- Dig deeper into equilibrium theory

- Get into market-microstructure: [**?**], [**?**]

- Dig into *emergent properties* of systems. Can they be formalized?

- Get into basics of Category-Theory [**?**] [**?**]

- Get into basics of Type-Theory (found good lectures on Youtube)

1.3   3rd Year: Finalizing, Publishing & Writing

I plan to be finished - or nearly finished - at the end of the 3rd year. In this year I will finalize the work of the 2nd year, publish the my main journal paper (and optional fun-papers if possible) and will write down the thesis.
To have a bit of distraction and to prevent myself to become too locked in in writing on the thesis I will also work on my optional fun-papers (see below) and hope to at least finish them and maybe publish them - at least I want to present them to 2-3 audiences (e.g. FP Lunch) to test the reaction (especially the Genesis-Paper).

- Finalize research of 2nd year

- Publish journal paper

- Write thesis

- Work on fun-papers

2 Papers

This is the list of papers I have in my mind and includes both work mandatory for the PhD and optional ones. The latter ones are just fun/philosophical-papers, not directly related to my PhD but somehow tangential with the very basic direction - they are intended to be worked on in my free-time and to free my head when wrestling too hard with my PhDs main work.

2.1 The Art of Iterating: Update-Strategies in Agent-Based Simulations

Type: Groundwork
Target: Conference/Journal
Requirement: Mandatory

When developing a model for an Agent-Based Simulation (ABS) it is of very importance to select the right update-strategy for the agents to produce the desired results. In this paper we develop a systematic treatment of all general properties, derive the possible update-strategies in ABS and discuss their interpretation and semantics something which is still lacking in the literature on ABS. Further we investigate the suitability of the three very different programming languages Java, Haskell and Scala with Actors to implement each of the update-strategies. Thus this papers contribution is the development of a new, general terminology of update-strategies and their implementation comparison in various kinds of programming languages.

2.2 Specification equals Code: An EDSL for pure functional Agent-Based Modelling & Simulation

Type: Groundwork
Target: Conference/Journal
Requirement: Mandatory

Building upon our previous work on update-strategies in Agent-Based Modelling & Simulation (ABM/S) where we showed that Haskell is a very attractive alternative to existing object-oriented approaches we ask in this paper if the declarative power of the pure functional language can be utilized to write specifications for simple ABM/S models which can be directly translated to our Haskell implementation.

2.3 Reactive Agents: Functional Reactive Programming and ABM/S

Type: Groundwork
Target: Conference/Journal

Requirement: Mandatory

In our previous work on update-strategies in Agent-Based Modelling & Simulation (ABM/S) we showed that Haskell is a very attractive alternative to existing object-oriented approaches but our presented approach was too limited and we hypothesized that embedding it within a functional reactive framework like Yampa would leverage it to be able to build much more complex models. In this paper we investigate whether this hypothesis is true by testing if our approach can easily be transferred to Yampa, what we really gain from it and if more complex models can become reality. As a proof-of-concept we build a large, complex model from Agent-Based Computational Economics (ACE) which simulates a whole economy.

2.4 Pure Functional ACE (Catchy title yet to be defined)

Type: Main PhD Work
Target: Journal
Requirement: Mandatory

Is the main work of the PhD and targeted at publication in a Journal. The exact topic and content will be clarified at the beginning of the 2nd year. Mainly it will describe how to implement Ionescus Framework of Gintis trading model and extend it to a more general Market-Model. It will also give an outlook on implementing it using dependent types.

2.5 Pure by Nature: A Library for pure Agent-Based Simulation & Modelling in Haskell

Type: Extension
Target: Conference
Requirement: Optional

This paper describes the ideas and theory behind the implementation of my ABM/S library "PureAgents" in Haskell.

2.6 Time in Games: a Tron Light-Cycle Game in Dunai

Type: Fun
Target: Conference
Requirement: Optional

This paper describes the 2D light-cycle game inspired by the movie Tron implemented in Dunai. It allows to turn back time.

2.7 Pure Functional Islamic Design

Type: Fun
Target: Conference
Requirement: Optional

Inspired by the paper "Functional Geometry" by Peter Henderson I had the idea to come up with a EDSL for declaratively describing pictures of islamic design which are then rendered using the gloss-library. From its focus totally unrelated to the PhD topic but still a great opportunity to learn Haskell, to learn to think functional, to learn to design my own EDSL - thus it may be a great paper to pursue even if I won't finish or produce something publishable.

2.8 The Genesis According to Computer-Science: Reality as Simulation of Free Will

Type: Philosophy
Target: ?
Requirement: Optional

I've always been interested in a deeper meaning behind things so I want to look into the philosophy and future of simulation: why do we simulate, what can we derive from simulations, what does it say that we humans simulate, what will the future of simulation be?
I claim that our ability to "simulate" in our mind separates our intelligence from those of the animals and that this is a unique property of humans. Also i think the future of simulation will be that humankind will do its own creation/live (artifical life, conciousness) which allows to accurately simulate a given setting - this of course could have ethical implications.

3 TODO-List as of March 6, 2017

1. EDSL for pure functional ABS/M: work on Functional Reactive ABS/M and develop EDSL

2. model checking and reasoning by quickcheck

3. implement SIRS

4. implement HAC

5. implement SugarScape

6. implement ACE Trading World

7. do literature research on MetaABS

8. work on theoretical background of MetaABS

9. Refine Research-Context A: ACE

   - look into ACE in general
   - look into agent-based market models
   - READ NASDAQ MARKET SIMULATION: seems extremely interesting as it builds an ABM for simulating the stock market.
   - look into ACE and formulate research questions. look for the killer-feature for ACE which is possible only in haskell and is not just a technical detail. EDSL for qualitative modelling? EDSL where model = runnable code?
   - NOT in economics but a METHOD IN COMPUTER SCIENCE
   - THUS: no research in combining complicated models (e.g. applying budish paper of batch-auctions to ABS model of markets): my direction agent-based market models IN GENERAL and what we can earn from doing that in FP
   - look for the killerfeature for ACE which is possible only in haskell and is not just a technical detail. EDSL for qualitative modelling? EDSL where model = runnable code?

10. Refine Research-Context B: Free Will in ABS

    - MetaABS: continue reading and thinking about Mind, Free Will and Conciousness for inspiration
    - Find a model to apply to: SugarScape?

11. Define 5 general research questions for each Research-Context

    - 2 related to FP
    - 1 related to integration of FP to ABM/S

- 2 related to ABM/S

12. 2nd Paper: functional reactive ABS

   - define formal agent definition
   - implement in yampa / dunai: sugarscape
   - define a minimal functional specification-language for ABM which is equals / very close to Haskell code
   - look into QuickCheck
   - problem: so far only agents with same static messagetypes, environment and states, can communicate: the agents are homogenous. how can we implement hetereogenous agents in this library?
   - implement wait blocking for a message so far. utilize yampas event mechanism?

13. Phd Thesis Title: Foundations of functional ABS

   - functional specification language
   - theory of computation in ABS: type theoretic and category theoretic foundations of ABS
   - implementing pure functional ABS/M: fusion with FRP
   - reasoning in functional ABS
   - a case study: pure functional ABS in ACE / Social Sciences / Artificial Life / Philosophy

14. got hint for publishing paper in: ppsm parallel problem solving in nature OR natural computation