Comparing Repast Java and Functional Reactive ABS

jonathan.thaler@nottingham.ac.uk

July 28, 2017

Abstract

This study we compares the Agent-Based Simulation programming libraries Repast and FrABS. As both are of fundamentally different programming paradigms - Repast uses Java, FrABS Haskell - both propagate fundamental different approaches in implementing ABS. In this document we seek to precicesly identify these fundamental differences, compare them and also look into general benefits and drawbacks of each approach.

Contents

0.1 Introduction

0.1.1 FrABS

0.1.2 Repast Java

0.1.3 Use-Cases

As use-cases on which we conduct the study we implement the following models in *both* libraries:

- JZombies - the 'Getting Started' example from Repast Java [1] - a first, very easy model, as there exists code-listing for Repast Java there is a 'standard' implementation, so comparison can be straight-forward and will serve as a first starting point.
- System-Dynamics SIR - study how the libraries deal with continuous time-flow.
- ABS SIR - study how the libraries support time-semantics.
- Sugarscape Model as presented in the book "Growing Artificial Societies - Social Sciences from the bottom up" by Joshua M. Epstein and Robert Axtell [1] - this highly complex model serves as a use-case for investigating how the libraries deal with a much more complex model with a big number of features. In this model there are no explicit time-semantics like in the SIR model: agents move in every time-step where the model is advanced in discrete time-steps.

0.1.4 Method

We don't compare library-features isolated but always directly using the use-cases as desribed above.

[1] `https://repast.github.io/docs/RepastJavaGettingStarted.pdf`

References

[1] Epstein, J. M., and Axtell, R. *Growing Artificial Societies: Social Science from the Bottom Up*. The Brookings Institution, Washington, DC, USA, 1996.