Using the Actor Model in Agent-Based Simulation: a Survey

Jonathan Thaler

October 25, 2016

Abstract

TODO: discrete SIR/S TODO: continuous SIR/S TODO: continuous bilateral trading according to gintis

1 Introduction

AKKA is nice but I think the actor model is not very well suited for simulations due to inherent concurrency where time is implicit. This makes simulations very difficult and also due to concurrency implementing a sync conversation among agents is very cumbersome. I have already experience with the Actor Model when implementing a small version of my Master-Thesis Simulation in Erlang which uses the Actor Model as well. For a continuous simulation it was actually not that bad but the problem there was that between a round-trip between 2 agents other messages could have already interfered - this was a problem when agents trade with each other, so one has to implement synchronized trading where only messages from the current agent one trades with are allowed otherwise budget constraints could be violated. Thus I think Erlang/Akka/Actor Model is better suited for distributed high-tolerance concurrent/parallel systems instead for simulations. Note: this is definitely a major point I have to argue in my thesis: why I am rejecting the actor model.
To put it another way: real concurrency (with threads) makes time implicit which is what one does NOT want in simulation. Maybe FRP is the way to go because it allows to explicitly model continuous and discrete time, but I have to get into FRP first to make a proper judgement about its suitability.
In the end what I want to achieve is a simulation-framework with explicit notion of time where discrete and continuous time is possible, which has an elegant EDSL included to write simulations/models (which will be then the same: model=simulation), allowing for reasoning about the model and the agent-based model itself implemented in a pure functional way.
real concurrency is not needed: simultaneous events can be modeled through explicit time but calculated sequential - when we reduce agents to process only one message after another and not multiple concurrently. thus true parallelism is only a technical detail for performance enhancement.

AKKA/PURE FUNCTIONAL: What if time is of no importance and only the continuous dynamics are of interest?
The real problem seems to be concurrency but i feel we can simulate concurrency by synchronizing to continuous time. computations are carried out after another but because time is explicitly modelled they happen logically at the same time. these rules hold: an agent cannot be in two conversations at the same time, the agent can be in only one or none conversation at a given time t.
new concept: not single, async messages, but syncronous conversations which (can) take time = multiple synchronous messages between agents which (can) change the state of an agent in the end.
AKKA: how can we simulate global time? how can we implement multistep conversations (by futures)?
AKKA upside: extreme huge number of agnts possible due to distributed and parallel technology
AKKA downside: depends on system & hardware: scheduler, system time, systime resolution (not very nice for scientific computation), much more complicated, debugging difficult due to concurrency, no global notion of time appart from systemtime, thus always runs in real-time, but there is no global notion of time in the actor model anyway, no EDSL full of technical details, no determinism, no reasoning

1.1 Hypothesis

AKKA: thus my prediction is: akka/actor model is very well suited to simulations which 1. dont rely on global time 2. dont have multi-step conversations: interactions among agents which are only question-answer. TODO: find some classical simulation model which satisfies these criterias.

2 The Actor Model

TODO: explain the actor model according to agha Agha (1986)

2.1 Erlang

TODO: erlang is an old implementation of the actor model

2.2 Akka

TODO: akka is a modern implementation of the actor model

3 Agents vs. Actors

Agents more a high-level concept, Actors low level, technical concurrency primitives

4 Implementing SIR/S

4.1 Modelling Time

4.1.1 Discrete Time

4.1.2 Continuous Time

5 Implementing continuous double-auctions

6 Conclusion

References

Agha, G. (1986). *Actors: A Model of Concurrent Computation in Distributed Systems*. MIT Press, Cambridge, MA, USA.