Benoit Gaudou
Jaime Simão Sichman (Eds.)

LNAI 9568

Multi-Agent-Based Simulation XVI

International Workshop, MABS 2015
Istanbul, Turkey, May 5, 2015
Revised Selected Papers

Springer

Lecture Notes in Artificial Intelligence 9568

Subseries of Lecture Notes in Computer Science

LNAI Series Editors

Randy Goebel
University of Alberta, Edmonton, Canada

Yuzuru Tanaka
Hokkaido University, Sapporo, Japan

Wolfgang Wahlster
DFKI and Saarland University, Saarbrücken, Germany

LNAI Founding Series Editor

Joerg Siekmann
DFKI and Saarland University, Saarbrücken, Germany

More information about this series at http://www.springer.com/series/1244

Benoit Gaudou · Jaime Simão Sichman (Eds.)

Multi-Agent-Based Simulation XVI

International Workshop, MABS 2015
Istanbul, Turkey, May 5, 2015
Revised Selected Papers

Springer

Editors

Benoit Gaudou
Capitole
University Toulouse 1
Toulouse Cedex 9
France

Jaime Simão Sichman
Escola Politécnica
Universidade de São Paulo
São Paulo
Brazil

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Artificial Intelligence
ISBN 978-3-319-31446-4 ISBN 978-3-319-31447-1 (eBook)
DOI 10.1007/978-3-319-31447-1

Library of Congress Control Number: 2016933472

LNCS Sublibrary: SL7 – Artificial Intelligence

© Springer International Publishing Switzerland 2016
This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.
The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.
The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made.

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG Switzerland

Preface

The 2015 edition of the Multi-Agent-Based Simulation (MABS) workshop was the 13th of a series that began in 1998. Its scientific focus lies in the confluence of social sciences and multi-agent systems, with a strong application/empirical vein, and its emphasis is on (a) exploratory agent-based simulation as a principled way of undertaking scientific research in the social sciences and (b) using social theories as an inspiration to new frameworks and developments in multi-agent systems.

The excellent quality level of this workshop has been recognized since its inception and its proceedings have been regularly published in Springer's *Lecture Notes in Artificial Intelligence* series. More information about the MABS workshop series may be found at http://www.pcs.usp.br/∼mabs.

MABS 2015 was hosted at AAMAS 2015, the 14th International Conference on Autonomous Agents and Multiagent Systems, which took place in Istanbul, Turkey, on May 5, 2015. In this edition, 22 submissions from 16 countries were submitted, from which we selected 12 for presentation (near 55 % acceptance). The papers presented in the workshop have been revised, and eventually extended and reviewed again, in order to make part of this post-proceedings volume.

We are very grateful to Frank Dignum, who gave a very inspiring invited talk, and to the participants, who provided a lively atmosphere of debate during the presentation of the papers and during the general discussion about the challenges that the MABS field faces. We are also very grateful to all the members of the Program Committee for their hard work. Thanks are also due to Michal Pechoucek and Longbing Cao (AAMAS 2015 workshop co-chairs), to Gerhard Weiss and Pınar Yolum (AAMAS 2015 general co-chairs), the last also having played the crucial role of AAMAS 2015 local organization chair.

January 2016

Benoit Gaudou
Jaime Simão Sichman

MABS 2015 Co-chairs

Organization

General and Program Chairs

Benoit Gaudou	IRIT, Toulouse 1 Capitole University, France
Jaime Simão Sichman	University of São Paulo, Brazil

MABS Steering Committee

Frédéric Amblard	IRIT, Toulouse 1 Capitole University, France
Luis Antunes	University of Lisbon, Portugal
Rosaria Conte	National Research Council, Italy
Paul Davidsson	Blekinge Institute of Technology, Sweden
Nigel Gilbert	University of Surrey, UK
Scott Moss	University of Koblenz-Landau, Germany
Keith Sawyer	University of North Carolina at Chapel Hill, USA
Jaime Simão Sichman	University of São Paulo, Brazil
Keiki Takadama	Tokyo Institute of Technology, Japan

Program Committee

Diana Francisca Adamatti	Universidade Federal do Rio Grande, Brazil
Alam Shah Jamal	Habib University, Pakistan
Frédéric Amblard	IRIT, Toulouse 1 Capitole University, France
Luis Antunes	University of Lisbon, Portugal
João Balsa	Universidade de Lisboa, Portugal
Carole Bernon	IRIT, Paul Sabatier University, Toulouse, France
Melania Borit	University of Tromsø, Norway
Philippe Caillou	INRIA, University of Paris Sud XI, France
Cristiano Castelfranchi	ISTC/CNR, Italia
Shu-Heng Chen	National Chengchi University, Taiwan
Paul Davidsson	Malmö University, Sweden
Frank Dignum	Utrecht University, The Netherlands
Virginia Dignum	Utrecht University, The Netherlands
Bruce Edmonds	Centre for Policy Modelling, UK
Francesca Giardini	Institute of Cognitive Sciences and Technologies, Italy
William Griffin	Arizone State University, USA
Francisco Grimaldo	Universitat de València, Spain
Laszlo Gulyas	AITIA International Informatics Inc., Hungary
David Hales	University of Szeged, Hungary
Rainer Hegselmann	University of Bayreuth, Germany
Marco Janssen	Arizona State University, USA

Satoshi Kurihara	Osaka University, Japan
Nicolas Marilleau	UMI UMMISCO IRD, France
Jean-Pierre Muller	CIRAD, France
Emma Norling	Centre for Policy Modelling, UK
Michael North	Argonne National Laboratory, USA
Paulo Novais	Universidade do Minho, Portugal
Juan Pavon Mestras	Universidad Complutense Madrid, Spain
Karoly Takacs	Corvinus University of Budapest, Hungary
Keiki Takadama	Tokyo Institute of Technology, Japan
Takao Terano	University of Tsukuba, Japan
H. Van Dyke Parunak	Soar Technology, USA
Natalie Van der Wal	VU University Amsterdam, The Netherlands
Daniel Villatoro	IIIA, Spain

Contents

Formalism Coupling in MABS

Extending the Gillespie's Stochastic Simulation Algorithm for Integrating Discrete-Event and Multi-Agent Based Simulation

Sara Montagna(✉), Andrea Omicini, and Danilo Pianini

DISI, Alma Mater Studiorum–Università di Bologna, via Sacchi 3, 47521 Bologna, Italy
{sara.montagna,andrea.omicini,danilo.pianini}@unibo.it

Abstract. Whereas Multi-Agent Based Simulation (MABS) is emerging as a reference approach for complex system simulation, the event-driven approach of Discrete-Event Simulation (DES) is the most used approach in the simulation mainstream. In this paper we elaborate on two intuitions: *(i)* event-based systems and multi-agent systems are amenable of a coherent interpretation within a unique conceptual framework; *(ii)* integrating MABS and DES can lead to a more expressive and powerful simulation framework. Accordingly, we propose a computational model integrating DES and MABS based on an extension of the Gillespie's stochastic simulation algorithm. Then we discuss a case of a simulation platform (ALCHEMIST) specifically targeted at such a kind of complex models, and show an example of urban crowd steering simulation.

Keywords: Multi-agent based simulation · Discrete-event simulation · Stochastic simulation · Gillespie algorithm · ALCHEMIST

1 Introduction

Computer simulation is a powerful tool used in principle in the analysis of real system behaviour in order to understand the main processes and mechanisms underlying the observed phenomena, and to predict the evolution of system dynamics under specified conditions. More recently, simulation became a common practice in the engineering of artificial computational systems, mainly during the design phase for testing, validating, and predicting system behaviour before actually starting the implementation phase [19].

Different approaches have been developed and proposed over the years to build models and execute simulations. In the last decade, Multi-Agent Based Simulation (MABS) became a reference approach for complex system simulation [4,15,17]. MABS provides constructs aimed at structuring the model following the multi-agent systems (MAS) paradigm, typically around three main abstractions – agent, society, and environment [15] –, and at defining their dynamic behaviour over time. MABS is usually introduced as a novel and alternative

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 3–18, 2016.
DOI: 10.1007/978-3-319-31447-1_1

approach to Discrete-Event Simulation (DES) and to Systems Dynamic (SD), which entered the simulation mainstream long ago.

When developing a MABS, the common practice is to build and execute the model on top of ad-hoc platforms that should provide the required tools to shape the agent-based model, as well as a scheduling module managing the simulation execution with respect to time. This paper focuses on such a module, by discussing the main approaches available in literature and implemented in the simulation engine of the most used platforms, such as NetLogo, Repast [24], MASON [14], Swarm [18], and AnyLogic: *(1)* continuous time; *(2)* discrete time with fixed interval, where time is advanced in equidistant steps; *(3)* discrete-event time, where time is advanced of discrete steps along a continuous timeline only once an event is triggered, which changes the state of the system. In particular we observe that the vast majority of the available simulation toolkits adopt a time-driven architecture, which is easier to be implemented both for the simulation developers and for the users when building a MABS, but has some problems of efficiency, accuracy, and coherence with the real system behaviour [16]. For this reason a time-driven architecture seems to bring in too many assumptions when modelling complex systems.

Following these preliminary remarks, we claim that an integration of MABS and DES could be crucial in the simulation of complex systems. Besides some recent literature [16], this also supported by a recent re-interpretation of EBS and MAS as two computational paradigms amenable of a coherent interpretation within a unique conceptual framework [25,26].

Along these lines, we here propose an event-driven stochastic computational model to build a full-fledged MABS engine and to structure agent-based models (ABM). Our scheduling module derives from two optimised versions [9,36] of the popular and successful Gillespie's stochastic simulation algorithm (SSA [10]), which was originally defined for the simulation of chemical systems and is intrinsically event driven. Our extension aims at making the algorithm more flexible so as to move from the pure chemistry towards the world of ABM. This impacts on the engine as well as on the ABM, here built around the concepts of reactions and compartments. To the best of our knowledge, our model is quite innovative in the MABS field.

Whereas we mostly aim at discussing a theoretical issue, rather than a new simulation framework, in this paper we present the case of a simulation platform (ALCHEMIST [30]) specifically developed for running such a kind of complex models. After quoting some successful application of our integrated DES/MABS model – in particular, in the fields of biology and pervasive computing – we show a running example of urban crowd steering to demonstrate its application in complex scenarios, and to test its computational performances.

2 Background

In the simulation mainstream, DES and MABS are usually presented as alternatives to continuous mathematical models – often referred as Systems Dynamic (SD) modelling approaches – and to Monte Carlo simulation.

2.1 Discrete-Event Simulation (DES)

DES is recommended when the dynamic of the system to be reproduced is characterised by a finite number of instantaneous events that are responsible for the changes in the system state. In between events, no change to the system is assumed to occur. Different events cannot be simultaneous. DES is usually very efficient since it allows to jump in time from one relevant event, corresponding to a state of the system, to the next one that modifies such a state. Thus it differs from SD, where the system state is assumed to change continuously over time.

Defining a DES means to model the behaviour of a system as an ordered sequence of non-continuous events, by specifying for each of them the *perturbations* in the system state it provokes, and the exact point in time when it has to be triggered. The simulation maintains at least one list of simulation events, sometimes called the *pending event set* since it lists those events that are pending as a result of previously simulated event but have not been simulated yet. Two functions should be defined: for choosing the next event from the list, and for changing the system state. Simulation proceeds by defining a clock that tracks time evolution, by advancing of discrete Δt once events occur.

2.2 Agent Based Modelling (ABM) and Simulation (MABS)

ABM and MABS are often discussed in the literature as alternative approaches to DES, and to mathematical models (such as differential equations), which model systems at the *macro-level*, *i.e.*, from the viewpoint of the entire population. On the contrary, ABM models the system at the *micro-level*, that is, at the level of its components: each active entity is modelled as an autonomous and interacting agent, situated in a dynamic environment with which it interacts by receiving inputs and by affecting its state with its actions. The global system-level behaviour of an ABM emerges as a result of the agent-to-agent interactions – *i.e.* from the societies of agents – and of the agent-to-environment interactions.

A computational engine for simulating agent behaviours, interactions, and environment – which may be spatially explicit and contain dynamic processes in addition to agents – is then needed to run the model—*i.e.* executing a MABS. Various platforms were developed for the purpose [2,33], which generally provide tools for developing, editing, and executing ABM, as well as for visualising the simulation dynamic. However, the crucial point here is to discuss the way they operate over a timeline, *i.e.* how the agents and environment behaviours are coupled and scheduled. We examine in depth such an issue in the next section.

3 Motivation

A crucial issue in MABS is how to deal with the evolution of time. The model for relating agent actions with the dynamics of the environment can be either *continuous* or *discrete*. Continuous approaches are rare and, in case, specifically used for modelling the endogenous dynamic of the environment coupled with discrete agent internal processes. In discrete approaches, time evolves with either regular intervals (time steps) or event executions (time is increased along a continuous timeline from one event to the next one) [17]. Among such techniques, the most widely used is the discrete approach with fixed time steps (hereafter called *time-driven*) [16,33]. The time-driven approach is the easiest one from both the simulation infrastructure perspective (no need to implement scheduler or event list, as required by DES) and the perspective of the modeller, who has just to specify the order of actions occurring during the same time step.

3.1 Time-Driven Drawbacks: A Comparison with DES

Efficiency — The time-driven approach is definitely less efficient than an event-driven approach: t has to pass by fixed time steps even if no actions are scheduled to be executed for changing the system state. Thus, modellers are required to choose the temporal granularity of actions: this normally corresponds to the fastest events, but it could obviously be a problem in those systems with a wide spectrum of time scales, such as in stiff systems possibly requiring an inefficient allocation of computational resources. In this regard, an event-driven approach can really improve the efficiency of a simulation by skipping those phases that are actually inactive.

Accuracy/Validity/Coherency — To be as close as possible to the MAS paradigm, agents should conduct their internal behaviour (actions and interactions) concurrently (so, simultaneous actions can occur), and be coupled with a possibly dynamic environment. In a time-driven approach, from time t, the state at $t + \Delta t$ results from combining all agent actions scheduled in the interval Δt with the environment evolution expected in the same interval. However, concurrency is lost: whereas agent and environment actions are interdependent, the order in which they are performed can radically change the overall result. Although some solutions were proposed (e.g., [17]), an event-driven approach is apparently the best compromise, since it limits the problem to those rare actions that are expected to be triggered simultaneously.

Congruence — The approximation of the reality in which all the entities of the system are updated simultaneously, as for the time-driven approach, often seems to be too far from the real behaviour of a complex system.

3.2 Related Work

The vast majority of the MABS platforms implement a time-driven scheduler. For instance, in NetLogo[1], one of the most widely used, time passes in discrete steps with the same length, called "ticks". However, MABS / DES integration was recently recognised as crucial: some of the most popular MABS platforms include an event-driven scheduler [16] (e.g. MASON [14] and Repast [24]). MASON is explicitly defined as "a fast discrete event multiagent simulation

[1] http://ccl.northwestern.edu/netlogo/.

library": agents are programmed to be *stepped*, *i.e.* their actions to be scheduled at some time in the future. To avoid conflicts, if multiple agents are scheduled for the same time, an ordering property can be specified. On the contrary, Repast mainly focuses on the time-driven approach, even though it provides constructs to possibly implement also event-driven simulations. The AnyLogic simulation software[2], which can boast hundreds of commercial and governmental organisations and hundreds of universities as users, still provides separate functionalities and tools for DES and MABS.

4 A Unified Conceptual Framework

The issue of a unified conceptual framework for MAS and EBS is indeed a complex one, and was generally faced in [25,26]. In the following, we just resume the main points of the framework that are relevant to the simulation problem—thus, to this paper.

4.1 MAS and Events

Generally speaking, the event-based architectural style has become prevalent for large-scale distributed applications, whereas MAS seemingly provide the most viable abstractions to deal with complex distributed systems. Thus, promoting a coherent integration of agent-based and event-based abstractions is generally relevant for the engineering of complex software systems, as demonstrated by the many approaches already integrating MAS with some sort of event-driven system:

- most of the agent architectures adopts some effective notion of event in order to provide for agent reactiveness—e.g., BDI architectures [34];
- most of the agent middlewares provide some event-based abstractions and mechanisms so as to deal with asynchronous message passing and environment change—such as JADE[3] [3] and TuCSoN[4] [29];
- [1] presents an agent-based architecture for coordinating event streams from WBSN
- ELDA is an event-driven agent meta-model [8];

4.2 MAS as EBS

The conceptual foundation of our unified framework lies in the observation that, from a software engineering viewpoint, agents and environment are the abstractions that represent the only *sources of events* in a MAS:

- agents represent the *designed* source of events, autonomously driving control towards their own *goals*, and producing *internal* events through their actions;

[2] http://www.anylogic.com.
[3] http://jade.tilab.com.
[4] http://tucson.unibo.it.

- environment models the *external* events that are relevant for the MAS, whose dynamics is in principle *unpredictable*, through abstractions of some sorts – environment resources, sensors & actuators, probes – that also capture the *diversity* of environment.

Along this line, an event-driven view of MAS is possible: MAS can be seen as event-driven systems, where

- agents encapsulate *internal events*—more generally, they encapsulate the activities in MAS that generate events according to some application criterion;
- environment models *external events* through dedicated abstraction—thus capturing the dynamics of the unpredictable events, which occur externally but are indeed relevant to the MAS.

Since agents encapsulate control along with the criteria for control – expressed in terms of high-level abstractions such as beliefs, goals, intentions, plans – articulated *events histories* can be modelled along with their *motivations* in MAS once they are interpreted as event-driven systems. Also, since environment in MAS is modelled as a first-class event-based abstraction, all *causes of change* and disruption in a MAS are modelled in a uniform way as *event pro-sumers* (that is, both producers and consumers of events).

Finally, social abstractions in MAS take care of coordinating multiple event flow according to their mutual dependencies in MAS – such as agent-agent and agent-environment dependencies – as in the case of *coordination artefacts* [27,28].

5 A Unified Stochastic Computational Model

In the following, we present a computational model derived from the popular and successful Gillespie's stochastic simulation algorithm (SSA [10]), and in particular from its notable and more efficient extensions developed by Gibson-Bruck [9] and by Slepoy [36]. Gillespie's SSA is a discrete and stochastic method intrinsically endowed with the event-driven properties [5]: introduced to model chemical systems, nowadays it represents the basis for many simulation platforms, in particular those aimed at the investigation of biochemical systems [6,11–13,22,32,37]. With significative extensions, it was recently used to model artificial systems inspired to chemical natural systems and ecology [23]. Thus, from the computational model perspective, the ABM we devised and presented in this paper is derived on Gillespie's well-accepted algorithm, especially because it intrinsically features event-driven properties, and it is also widely used in the simulation of complex systems. Moreover, the possibility to adopt the chemical metaphor is apparently the most natural way to model those systems featuring self-* properties [7]. From the perspective of the simulation engine, the optimised extensions of the Gillespie's SSA [9,36] are very efficient, and make it possible really fast simulation runs.

5.1 Gillespie's SSA as an Event-Driven Algorithm

Gillespie's SSA revolves around the idea of modelling a chemical system as a single space filled with molecules that interact through a number of reactions defining how they combine. The instantaneous speed of a reaction is called *propensity*, and depends on the kinetic rate of the reaction as well as on the number of molecules of all the reagents involved. For a reaction i with k reactants, j products, rate r of the form $R_0 + R_1 + \ldots + R_k \xrightarrow{r} P_0 + P_1 + \ldots P_j$, the propensity a_i is defined as: $a_i = r \cdot [R_0] \cdot [R_1] \cdot \ldots \cdot [R_k]$ where $[R_a]$ is the number of molecules of species R_a. Given that, the algorithm relies on the idea that the system can be simulated by effectively executing the reactions one by one, and by changing the system status accordingly.

The algorithm follows four steps: *(i)* select the next reaction μ to be executed; *(ii)* calculate the time of occurrence of μ according to an exponential time distribution and make it the current simulation time; *(iii)* change the environment status in order to reflect this execution; *(iv)* update the propensities of the reactions. Such algorithm is event-driven in that it executes only one reaction/event at a time: it changes the state of the system, and consequently the event list.

5.2 Gillespie's SSA in MABS: Related Work

Most of the few works integrating the Gillespie's SSA into a MABS refer to the simulation of living organisms, which naturally manifest stochastic behaviour. Integration may occur at different levels: the SSA can either model the agents and environment behaviour (which is the sort of integration we are aiming at) or only a portion of the system behaviour. In [40], for instance, only the diffusion of molecules is modelled, whereas molecules, in turn, are modelled as agents with trivial behaviours. A more advanced method is presented in [35], where a multicellular system is modelled as a MAS whose agents are cells whose behaviour is modelled as a set of reactions, simulated via a modified version of the SSA, interacting with the local environment, populated by diffusing molecules.

5.3 Gillespie's Optimised Versions

The algorithm was improved by various works in literature. In particular, both [9] and [36] optimise the base algorithm in two phases: the selection of the next reaction to execute, and the update of the reaction pool – pending event list – once an event has been executed. For the latter, they both rely on the concept of *dependency graph*, namely, a data structure statically linking each reaction to the set of reactions whose execution speed may be influenced by the execution of the former. Even though the optimisation does not affect the worst case complexity, it offers great benefits in the average case, since most of the reactions are not interdependent—which actually impacts optimisation the most [36].

The selection of the next reaction to execute is where those improved algorithms differ most. The "Composition-Rejection" (CR) algorithm [36] smartly groups reactions with similar rate in such a way that the number of such groups remains constant for most biological scenarios. Once reactions are grouped, the algorithm can correctly choose the next one to execute in constant time: if the number of groups is constant throughout the simulation, the algorithm is actually able to select the next reaction in constant time. The same selection operation in [9] is made by computing a putative execution time for each reaction, and then using a binary heap data structure to sort them. This way, the next reaction to execute is always the root node of the tree, and the selection is performed in constant time. However, once the reaction has been executed, all the dependent reactions must be updated and re-sorted in the tree. In the worst case, this takes logarithmic time with respect to the number of reactions.

5.4 Bridging DES and MABS Through an Enhanced Gillespie's SSA

Based on [9,36], we here adopt and extend a novel stochastic computational model [30] for *(i)* integrating into a MABS toolkit a DES scheduler based on the Gillespie's SSA, and *(ii)* specifying the agent (internal and interacting) and environment behaviours by the concept of reactions, properly extended from the original concept of chemistry found in Gillespie.

The Simulation Engine. Both the Gibson-Bruck's [9] and Slepoy's [36] schedulers are granted to correctly simulate a Poisson process. However, in order to build a full-fledged DES engine, we must offer the possibility to schedule also non-Markovian events. For instance, let us consider the simulation of an agent that acts at every fixed time interval (e.g. a walking man): this, clearly, is not a memoryless process. Gibson-Bruck offers a more suitable base for a generic DES: in fact, its selection mechanism for next reaction is orthogonal to the way the putative times are computed. This allows the neat separation between the generation of times (namely, the time distribution of each event) and the actual scheduling (choice of which event should run next). Moreover, Slepoy's grouping fails if there are instantaneous reactions (for which $r=\infty$), and there is no straightforward way to pick non-markovian events using CR.

Another feature needed to shift the paradigm from pure chemistry towards ABM is the possibility to simulate multiple, separate, interacting, and possibly mobile entities. This can be partly addressed by the notion of intercomunicating compartments [6,11,22,37], in a way that makes it possible also to model systems characterised by a set of connected volumes and not just a unique chemical solution. The hardest challenge in simulating multiple compartments with an efficient SSA is in improving the dependency graph: reactions that could be interdependent but occur in separated compartments should not be marked as interconnected within the dependency graph. Mobility makes everything even harder, since it may lead to the creation and disruption of communication channels between compartments, and consequently to simulation-time changes in the structure of such dependency graph. Summarising, supporting dynamic environments with multiple mobile compartments requires the dependency graph to become a dynamic data structure, which cannot be pre-computed at the simulation initialisation and kept static.

A possibility for efficiently adapting a dependency graph to a network of compartments could be to define the input and output contexts for each reaction, namely, the places where reactions respectively "read" their reactants and "write" their products. Multiple contexts could be defined: in [30] we propose the adoption of three levels: `local`, `neighborhood` and `global`, along with suitable procedures to correctly compute the graph's edges. On top of this finer-grain locality concept, if the model supports compartment mobility, the dependency graph must support the dynamic addition and removal of reactions.

The Computational Model. Such an engine, despite its improved flexibility, is still bound to a world made of molecules, reactions, and compartments. Typical ABM require instead the specification of higher-level concepts, such as agent, internal and interactive behaviour, and environment.

Here, then, a compartment can represent an agent, and all the possible events in the model should in the end break down to a set of reactions (see Fig. 1). In the following discussion, we interchangeably use compartment/agent/node and reaction/events as synonyms.

As a first step, we introduce the notion of *environment as a first-class abstraction* [38], missing in chemistry-derived SSA. The environment has the responsibility to provide, for each compartment, a set of compartments that are its neighbours. The function of the current environment state that determines whether or

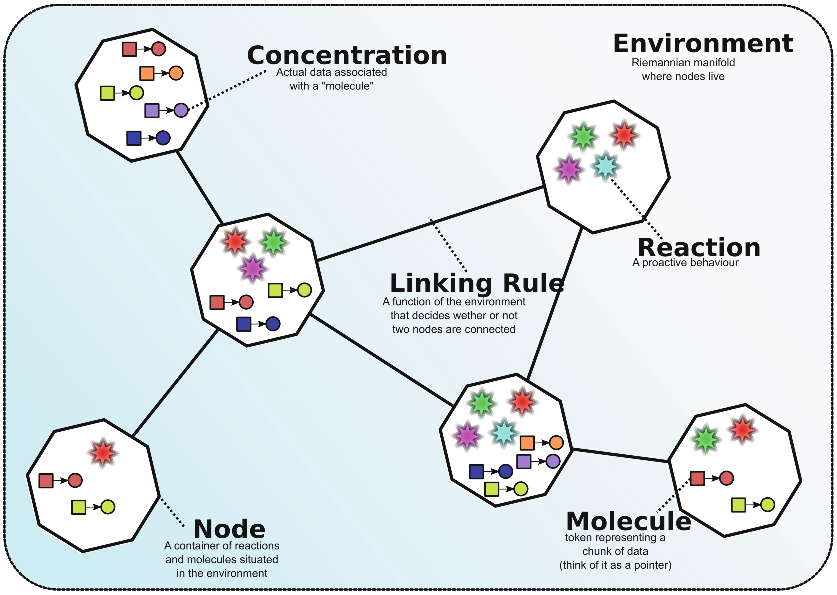

Concentration
Actual data associated with a "molecule"
Environment
Riemannian manifold where nodes live
Reaction
A proactive behaviour
Linking Rule
A function of the environment that decides wether or not two nodes are connected
Node
A container of reactions and molecules situated in the environment
Molecule
token representing a chunk of data (think of it as a pointer)

Fig. 1. The computational model

not two nodes are neighbours can be arbitrarily complicated. Also, it is responsible of exposing possible physical boundaries, namely, to limit movements of compartments situated within the environment.

The fact that reactions are the only abstraction the modeller can rely upon in order to let the simulated system progress does not really hinder expressivity. In fact, our model does not use a strictly-chemical agent behavioural model [39]: instead, it generalises the concept of reaction as "a set of conditions that, when matched, trigger a set of actions on the environment". Accordingly, a condition is a function that associates a numeric value ranging from zero to positive infinity to each possible state of the environment. If such value is zero, the event can not be scheduled; otherwise, it is up to the reaction to interpret the number. In this framework, actions are arbitrary changes to the environment. To allow the dependency graph to be built, both conditions and actions should expose the set of possible data items (molecules) that they may read or modify. Also, both conditions and actions should expose a context of the type `local`, `neighborhood` or `global`, used to determine the input and output contexts for the reaction itself.

Reactions compute their own expected execution time. Such putative time is generated by a function ("rate equation" in the following) taking as input all the values generated by conditions and a time distribution. The engine may require putative time to be updated in case that *(i)* the reaction has been executed, or *(ii)* another reaction which the reaction depends on has been executed.

The injection of a time distribution as a parameter for the rate equation makes it possible to model a whole set of events that are not exponentially distributed: imagine, for instance, a burette dropping some quantity of reactant in a compartment every fixed time interval. Such an event is clearly not memoryless, and can be modelled in the proposed framework with a reaction having no condition and a single action increasing the reactant concentration in the compartment. Such a reaction can be fed with a Dirac comb, and the resulting system would seamlessly mix exponential and non-exponential events.

The low expressive power of the classical concentration is probably the hardest challenge when trying to extend a chemical-born meta-model towards the rich world of agents. To this end, we define *concentration* as depending on the actual meta-model: so, in the following, concentration is related to the "the data items agents can manipulate". Besides the trivial example of chemistry, where data items are integer numbers, let us consider a distributed tuple spaces model: in this case, the molecules would be tuples, and the concentration would be defined as "the number of tuples matching a certain tuple". Clearly, such flexibility comes at a cost: since the conditions and actions operating on different concentrations dramatically change their behaviour, for any possible class of data items the meta-model must be instanced with a proper set of conditions and actions that can act on such a sort of concentration.

We argue that our model allows for enough flexibility to support complex agents, too: for instance, a complex mind cycle could be implemented as an action, and a condition expressing whether or not the agent is alive would allow it to run. In Sect. 6 some evidence for our assertion is provided, by referencing works that rely on such computational model to build and simulate complex scenarios.

6 ALCHEMIST as a Stochastic DES/MABS Platform

ALCHEMIST[5] [30] is a simulation platform implementing the simulation algorithm described in Subsect. 5.4, thus allowing ABM to be built according to the computational model described in the same Subsection. ALCHEMIST is a highly-performant simulator faster than advanced ABM simulators such as Repast [30], able to load simulation specifications from XML files, which are normally generated using domain-specific languages (DSL). ALCHEMIST comes distributed with several DSL, one of which was explicitly designed to model pervasive service ecosystems [23].

The framework was already used in literature producing promising results, thus demonstrating the feasibility of SSA-based ABM. Most such works simulate pervasive computing scenarios (e.g. [20,31]), but the same framework has been used to model multicellular biological systems in [21].

Besides using the previously described engine and model, ALCHEMIST comes with a set of features valuable to the modeller, such as:

- support for complex environments, which can be generated automatically upon black and white images;
- loading of maps data from OpenStreetMap, supporting (through Graphhopper) navigation of users, cars and bicycles along proper roads;
- loading of GPS traces;
- advanced mobility models for agents, ranging from interpolation of traces using maps data to realistic indoor pedestrians behaviour.

6.1 A Case Study: Crowd Steering in London

To demonstrate how our computational model can be applied to complex scenarios, and also to show how the simulation algorithm performs, we here present one notable application example of ABM: crowd steering at the urban scale. The idea is to guide people in an urban area towards locations they are interested in, following the most satisfactory path according to any arbitrary metric—for instance, the shortest, the quickest, the less trafficked, or the less polluted. Such situations may vary in time and space, dynamically changing the emergent suggested path. Moreover, according to the ABM approach, system has to evolve without any centralised computing system involved, and based on the local interactions among agents and agent-environment—namely, in a self-organising fashion.

[5] http://alchemist.apice.unibo.it.

Model. We set up our model in the city of London. According to the ABM abstractions, we model the *environment* as a static network of nodes randomly deployed along the city streets. Such nodes must be equipped with network capabilities similar to those of a Wi-Fi access point; in particular, they must allow nearby mobile devices to connect to all the static nodes within their communication range. We suppose users, modelled as *agents*, to be equipped with smart devices aware of their position. We think such hypotheses are quite realistic: nowadays, almost everybody brings (at least) one smart device along with her, and such devices are always equipped with a GPS positioning system and networking capabilities.

Each environmental node is responsible of computing – according to internal reactions – the local value of some distributed data structures, called gradients [7], relying only on values provided by neighbouring nodes and on contextual information. Each agent receives the gradient value from each environmental node in its neighbourhood. Such values are compared and the lowest is chosen as current destination. The agent is instructed to navigate pedestrian roads towards the current destination, following the navigation system suggestions.

Simulation. Our simulation consists of 1000 nodes deployed along the city streets, with a communication range of 400m. We mark as unfavourable the area comprised between Westminster bridge and Golden Jubilee Bridge, and as particularly attractive the area south of Blackfriars Bridge. In Elephant and Castle, a point of interest (POI) propagates a gradient. Such a data structure is dynamic, and as such it is computed by simulating the corresponding reactions.

In our setup, 30 pedestrians located in the proximity of St. James Park want to be navigated towards the POI. In order to do so, they follow down the gradient. Being the unfavourable area along the shortest path, some of the nodes are steered south and some north. Such a group division is completely emergent. Simulation snapshots are shown in Fig. 2.

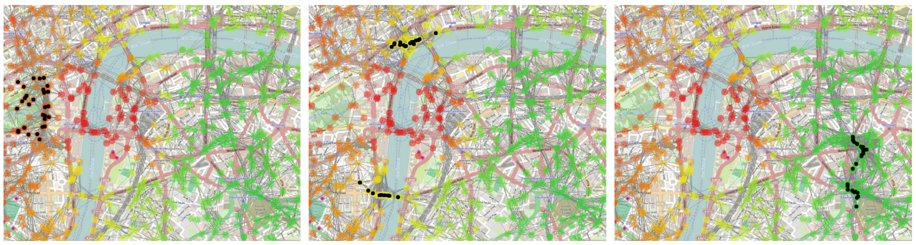

(a) Initial (b) Ongoing (c) Near-end

Fig. 2. Three snapshots of the simulation. Our altered distance metric (the gradient) goes from green (most favourable and close to destination) to red (unpleasant or far from destination). Agents (in black) climb this structure towards greener values (Color figure online).

The simulator completed the whole simulation (863450 simulated seconds) in around 580 s with the graphical interface attached on an Intel Core i5–2500 with 8 GB of RAM running Linux 3.17. The dependency graph for such experiment counted 2088 nodes, while the number of arcs ranged from 30520 to 35811. The number of nodes did not change because no new nodes, no new events were injected while the simulation was running, nor any was removed. The number of arcs, instead, varied because of the agent movement across the map, which led to changing neighbourhood relationships.

7 Conclusion and Future Work

In this paper we adopt an extension to the Gillespie's SSA as a stochastic event-driven algorithm that lead us to a novel model enabling the integration of DES and MABS. The model impacts both on the way the scheduling module of a MABS platform is implemented, and on the way agents and environment behaviours have to be defined. Since it draws its inspiration from the chemistry word, it is grounded on the concepts of chemical reactions and biological compartments. However, such abstractions are suitably extended to properly fill the gap with the abstractions characteristic of an ABM and to be most suitable for the simulation of a more generic scenario. To the best of our knowledge this is one of the first attempts to bring Gillespie's SSA into the MABS scientific field.

Even though the meta-model we envision is inspired by chemical systems, and its main applications are actually devoted to reproduce the dynamic of biochemical or ecological systems, self-organising systems, or artificial systems with an inspiration into ecology, it has the potential to be actually adapted to model other sorts of systems featuring suitable description in terms of atomic, instantly-happening events. The three main concepts of environment, reaction, and concentration are those that have to be suitably extended to move towards a higher level of expressiveness. In particular, in this work we show how the notions of environment and reaction can be made generally more expressive. Concentration requires instead a definition tailored on the specific scenarios, since its meaning in our framework is generically "the kind of data that is manipulated". However, it is worth mentioning that our intent is not to provide an ABM generally valid for any application domain. For instance we believe that for those systems manifesting elaborated cognitive abilities the chemical abstraction, even if opportunely extended as we presented here, might not be sufficient.

Finally, we discuss ALCHEMIST as a simulation framework built upon those concepts, which demonstrates the feasibility and expressiveness of our DES/MABS integrated approach through a simulation of crowd steering at the urban scale.

In future, we intend to measure the impact of an event-based scheduling both on the quality of the simulation outcome and on the execution time. Our expectation is that an event-driven algorithm will provide a the same simulation precision of an extremely fine grained time-driven algorithm with much higher performance, due to the number of "empty" events ignored. Also, we expect more coarse time ticks to progressively increase the simulation performance, at expense of precision in the simulation outcome.

References

1. Aiello, F., Bellifemine, F.L., Fortino, G., Galzarano, S., Gravina, R.: An agent-based signal processing in-node environment for real-time human activity monitoring based on wireless body sensor networks. Eng. Appl. Artif. Intell. **24**(7), 1147–1161 (2011)
2. Allan, R.: Survey of agent based modelling and simulation tools. Technical report DL-TR-2010-007, Computational Science and Engineering Department (2010)
3. Bellifemine, F.L., Caire, G., Greenwood, D.: Developing Multi-Agent Systems with JADE. Wiley, Chichester (2007)
4. Bonabeau, E.: Agent-based modeling: methods and techniques for simulating human systems. Proc. Nat. Acad. Sci. **99**(s. 3), 7280–7287 (2002)
5. Burrage, K., Burrage, P.M., Leier, A., Marquez-Lago, T., Nicolau Jr., D.V.: Stochastic simulation for spatial modelling of dynamic processes in a living cell. In: Koeppl, H., Setti, G., di Bernardo, M., Densmore, D. (eds.) Engineering Approaches to Systems and Synthetic Biology, pp. 43–62. Springer, New York (2011)
6. Ciocchetta, F., Guerriero, M.L.: Modelling biological compartments in Bio-PEPA. ENTCS **227**, 77–95 (2009)
7. Fernandez-Marquez, J.L., Serugendo, G.D.M., Montagna, S., Viroli, M., Arcos, J.L.: Description and composition of bio-inspired design patterns: a complete overview. Nat. Comput. **12**(1), 43–67 (2013)
8. Fortino, G., Garro, A., Mascillaro, S., Russo, W.: Using event-driven lightweight DSC-based agents for MAS modelling. Int. J. Agent-Oriented Softw. Eng. **4**(2), 113–140 (2010)
9. Gibson, M.A., Bruck, J.: Efficient exact stochastic simulation of chemical systems with many species and many channels. J. Phys. Chem. A **104**(9), 1876–1889 (2000)
10. Gillespie, D.T.: Exact stochastic simulation of coupled chemical reactions. J. Phys. Chem. **81**(25), 2340–2361 (1977)
11. González Pérez, P.P., Omicini, A., Sbaraglia, M.: A biochemically-inspired coordination-based model for simulating intracellular signalling pathways. J. Simul. **7**(3), 216–226 (2013)
12. Hoops, S., Sahle, S., Gauges, R., Lee, C., Pahle, J., Simus, N., Singhal, M., Xu, L., Mendes, P., Kummer, U.: Copasi - a complex pathway simulator. Bioinformatics **22**(24), 3067–3074 (2006)
13. Kierzek, A.M.: STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm. Bioinformatics **18**(3), 470–481 (2002)
14. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.C.: Mason: a multi-agent simulation environment. Simulation **81**(7), 517–527 (2005)
15. Macal, C.M., North, M.J.: Tutorial on agent-based modelling and simulation. J. Simul. **4**, 151–162 (2010)
16. Meyer, R.: Event-driven multi-agent simulation. In: Grimaldo, F., Norling, E. (eds.) MABS 2014. LNCS, vol. 9002, pp. 3–16. Springer, Heidelberg (2015)
17. Michel, F., Ferber, J., Drogoul, A.: Multi-agent systems and simulation: a survey from the agents community's perspective. In: Multi-Agent Systems: Simulation and Applications. CRC Press (2009)

18. Minar, N., Burkhart, R., Langton, C.: The Swarm simulation system: a toolkit for building multi-agent simulations. Technical report 96–06-042, Santa Fe Institute (1996)
19. Molesini, A., Casadei, M., Omicini, A., Viroli, M.: Simulation in agent-oriented software engineering: the SODA case study. Sci. Comput. Program. **78**(6), 705–714 (2013)
20. Montagna, S., Pianini, D., Viroli, M.: Gradient-based self-organisation patterns of anticipative adaptation. In: 6th IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2012). pp. 169–174 (2012)
21. Montagna, S., Pianini, D., Viroli, M.: A model for Drosophila Melanogaster development from a single cell to stripe pattern formation. In: 27th Annual ACM Symposium on Applied Computing (SAC 2012). pp. 1406–1412. ACM (2012)
22. Montagna, S., Viroli, M.: A framework for modelling and simulating networks of cells. ENTCS **268**, 115–129 (2010)
23. Montagna, S., Viroli, M., Fernandez-Marquez, J.L., Di Marzo Serugendo, G., Zambonelli, F.: Injecting self-organisation into pervasive service ecosystems. Mob. Netw. Appl. **18**(3), 398–412 (2013)
24. North, M., Collier, N., Ozik, J., Tatara, E., Macal, C., Bragen, M., Sydelko, P.: Complex adaptive systems modeling with Repast Simphony. Complex Adapt. Syst. Model. **1**(1), 3 (2013)
25. Omicini, A.: Event-based vs. multi-agent systems: towards a unified conceptual framework. In: 2015 19th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD 2015). IEEE Computer Society (May 2015)
26. Omicini, A., Fortino, G., Mariani, S.: Blending event-based and multi-agent systems around coordination abstractions. In: Holvoet, T., Viroli, M. (eds.) Coordination Models and Languages. LNCS, vol. 9037, pp. 186–193. Springer, Heidelberg (2015)
27. Omicini, A., Ricci, A., Viroli, M.: Coordination artifacts as first-class abstractions for MAS engineering: state of the research. In: Garcia, A., Choren, R., Lucena, C., Giorgini, P., Holvoet, T., Romanovsky, A. (eds.) Software Engineering for Multi-Agent Systems IV: Research Issues and Practical Applications. LNCS, vol. 3914, pp. 71–90. Springer, Heidelberg (2006)
28. Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., Tummolini, L.: Coordination artifacts: environment-based coordination for intelligent agents. In: Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M. (eds.) 3rd international Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004). LNCS, vol. 1, pp. 286–293. ACM, New York, USA (2004)
29. Omicini, A., Zambonelli, F.: Coordination for internet application development. Auton. Agents Multi-Agent Syst. **2**(3), 251–269 (1999)
30. Pianini, D., Montagna, S., Viroli, M.: Chemical-oriented simulation of computational systems with Alchemist. J. Simul. **7**(3), 202–215 (2013)
31. Pianini, D., Viroli, M., Zambonelli, F., Ferscha, A.: HPC from a self-organisation perspective: the case of crowd steering at the urban scale. In: High Performance Computing Simulation (HPCS 2014). pp. 460–467 (2014)
32. Priami, C., Regev, A., Shapiro, E., Silverman, W.: Application of a stochastic name-passing calculus to representation and simulation of molecular processes. Inf. Process. Lett. **80**, 25–31 (2001)
33. Railsback, S.F., Lytinen, S.L., Jackson, S.K.: Agent-based simulation platforms: review and development recommendations. Simulation **82**(9), 609–623 (2006)

34. Rao, A.S., Georgeff, M.P.: Modeling rational agents within a BDI architecture. In: Allen, J.F., Fikes, R., Sandewall, E. (eds.) 2nd International Conference on Principles of Knowledge Representation and Reasoning (KR'91), pp. 473–484. Morgan Kaufmann Publishers, San Mateo, CA (1991)
35. Shimoni, Y., Nudelman, G., Hayot, F., Sealfon, S.C.: Multi-scale stochastic simulation of diffusion-coupled agents and its application to cell culture simulation. PLoS ONE **6**(12), e29298 (2011)
36. Slepoy, A., Thompson, A.P., Plimpton, S.J.: A constant-time kinetic Monte Carlo algorithm for simulation of large biochemical reaction networks. J. Chem. Phys. **128**(20), 205101 (2008)
37. Versari, C., Busi, N.: Efficient stochastic simulation of biological systems with multiple variable volumes. ENTCS **194**(3), 165–180 (2008)
38. Weyns, D., Omicini, A., Odell, J.: Environment as a first-class abstraction in multi-agent systems. Auton. Agents Multi-Agent Syst. **14**(1), 5–30 (2007)
39. White, T., Pagurek, B.: Towards multi-swarm problem solving in networks. In: Proceedings of International Conference on Multi Agent Systems 1998, 333–340 (1998)
40. Zabet, N.R., Adryan, B.: GRiP: a computational tool to simulate transcription factor binding in prokaryotes. Bioinformatics **28**(9), 1287–1289 (2012)

Coupling Micro and Macro Dynamics Models on Networks: Application to Disease Spread

Arnaud Banos[1], Nathalie Corson[2], Benoit Gaudou[3](✉), Vincent Laperrière[4], and Sébastien Rey Coyrehourcq[5]

[1] UMR Géographie-cités, CNRS, University of Paris 1 Panthéon Sorbonne, University of Paris 7 Paris Diderot, Paris, France
benoit.gaudou@ut-capitole.fr
[2] LMAH, University Le Havre, Normandie University, FR CNRS 3335, Le Havre, France
[3] UMR 5505 IRIT, CNRS, University of Toulouse, Toulouse, France
[4] UMR ESPACE, CNRS, University Nice Sophia Antipolis, Avignon University, Aix Marseille University, Nice, France
[5] UMR IDEES, CNRS, University of Rouen, Rouen, France

Abstract. A hybrid model coupling an aggregated equation-based model and an agent-based model is presented in this article. It is applied to the simulation of a disease spread in a city network. We focus here on the evaluation of our hybrid model by comparing it with a simple aggregated model. We progressively introduce heterogeneities in the model and measure their impact on three indicators: the maximum intensity of the epidemic, its duration and the time of the epidemic peak. Finally we present how to integrate mitigation strategies in the model and the benefits we can get from our hybrid approach over single paradigm models.

Keywords: Hybrid model · ODE · Metapopulation · Network · Disease spread

1 Introduction

The modelling of socio-environmental processes often requires to couple processes defined at distinct temporal and spatial scales. Models involving a single paradigm to describe all the processes, such as either an aggregate approach at the macroscopic scale (with a system dynamic approach) or a totally disaggregated microscopic approach (with an agent-based approach), fail to describe multi-scale phenomena. The aim of this article is thus to introduce a hybrid model coupling micro and macro dynamics models and to assess step-by-step the impact of the heterogeneity and the stochasticity introduced by microscopic model.

As a case study, we consider a disease spread in a network of cities. The two main dynamics we take into account are the disease spread within each city and its spread between cities through air traffic. The former dynamics involves

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 19–33, 2016.
DOI: 10.1007/978-3-319-31447-1_2

a large number of people in each city (millions of people) and its temporal scale order is of several weeks. In contrarily the latter one involves few (hundreds) of people in each airplane, and a flight temporal scale order is of several hours.

The heterogeneities that will be studied in the model are related to the initial population distribution, heterogeneity in the network (we will experiment various network topologies) and heterogeneity in flight duration to take into account the space size.

The classical approach to model such phenomena is named *metapopulation approach*, comes from ecology [8] and has been applied to various fields and in particular to disease spread in large-scale networks [1,3,11]. The main idea is that each node of the network has a dynamic described using a system dynamics approach (often described using an Ordinary Differential Equation (ODE) system). In addition edges represent migration (modelled as instantaneous streams) between nodes. Such an approach allows modellers to study disease spread conditions and to test various spread mitigation strategies [11].

We argue that this approach is too limited to take into account strategies dealing with individual behaviours. In this article we thus go one step further by introducing individual and possibly heterogeneous passengers in order to better take into account intentionality, reflexivity and adaptability of human beings [4,5]. Due to space limitation, we cannot present this in details; we can only give an insight of the benefits of our approach. The main contribution is to design, build and evaluate a frame model that will be extend in the future with more complex individual behaviors.

The article is organized as follows. Section 2 introduces the various modelling approach used in this article. Section 3 presents the model and Sect. 4 the method we use to evaluate the hybrid model. Section 5 presents the results and Sects. 6 and 7 introduces mitigation strategies in the model and conclude.

2 State of the Art

2.1 System Dynamics Modeling of Epidemic

One of the traditional ways to describe the dynamic of systems is the *system dynamics* approach [6], *i.e.* the description of the evolution of macroscopic variables using a set of equations, often an ODE system. Generally it is not possible to get an exact solution of such system, and a numerical integration method (*e.g.* Runge Kutta [13]) is used to approximate it. Nevertheless we can prove analytical properties on equilibrium points and on their stability. In epidemiology, the most famous equation-based model is the SIR model [9]. It considers the global population of N people as a whole and describes the evolution of the 3 stocks of Susceptible (S), Infected (I) and Recovered (R) people using the system presented in Fig. 1 ($N = S + I + R$ in this system). The numerical integration using the Runge Kutta 4 (RK4) method is plotted in Fig. 2.

The parameter β is the infection rate of the disease in case of contact between a Susceptible and an Infected individual, *i.e.* it describes the transition between Susceptible and Infected stocks. Similarly α describes the transition between

$$\begin{cases} \dfrac{dS}{dt} = -\dfrac{\beta IS}{N} \\ \dfrac{dI}{dt} = \dfrac{\beta IS}{N} - \alpha I \\ \dfrac{dR}{dt} = \alpha I \end{cases} \quad (1)$$

Fig. 1. ODE system for a SIR model

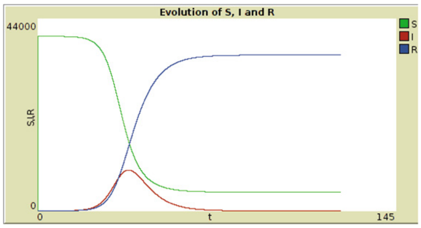


Fig. 2. Plot of the evolution of S, I and R for a SIR model. We choose: $\alpha = 0.2$, $\frac{\beta}{N} = 0.5$.

the Infected and the Recovered states, *i.e.* it represents the recovery rate. This system can be graphically drawn using a Forrester diagram [6] as in Fig. 3.

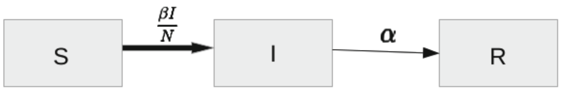


Fig. 3. SIR compartiments

This system is very simple but can be complexified by introducing additional stocks or streams between stocks. For example if people that have recovered from the disease can be infected again, a stream between the Recovered and Susceptible stocks can be added (we get the SIRS model). If the disease is characterized by a time period when the individual is exposed to the disease but not yet seek, an additional stock (Exposed) can be added between Susceptible and Infected, we thus get the SEIR model.

2.2 Metapopulation Models

One of the strongest limitation of the system dynamic approach is that it considers the population as a whole. It has been extended in ecology by the so called *metapopulation* approach [8]: the aim is to represent several populations and the migration relationships between them. The whole population is thus split into several populations that are the nodes of a graph, the edges representing the possible migrations. The dynamic inside each node (population) is described by an ODE model whereas migrations are managed as instantaneous streams between nodes. This approach has also been applied in epidemiology [3,11] to deal with the spread of a disease over a network of cities.

The metapopulation approach allows modellers to study how to control an epidemic by testing some mitigation strategies at the city level, such as the quarantine (a city is closed for arrivals and departures), avoidance (the traffic avoids a city, but airplanes can still leave it) or at the individual level, such as the risk culture (people being aware they are infected can choose to postpone their travel in order to avoid to infect other persons). Nevertheless such strategies remain limited as they can not take into account heterogeneous individual travels: individuals are not and cannot be represented in such models. This is the main feature of the agent-based modelling approach.

2.3 Agent-Based Models

The main idea of the agent-based models is to describe individual entities of a system and to let emerge the macroscopic expected behavior thanks to the interactions among entities. In epidemiology, it has also been successfully applied in studying the diffusion of mumps in Portugal [14] or plague in Madagascar [10].

Works such as [12] or [2] have already investigated the link between agent-based and equation-based models, but their point was to compare the two approaches in their representation of the same phenomenon. They highlight the difference of paradigm, scale of representation and way of thinking the phenomenon. But as far as we are aware, few articles investigate deeply the coupling of these two approaches into one single model as we do in the sequel.

3 A Micro-Macro Coupled Model: The MicMac Model

In this section, we present the MicMac coupled model. It has been developed using the Netlogo platform [16] (*c.f.* Fig. 4) dynamically coupled with a Scala extension we designed for Runge-Kutta 4 numerical integration method. Due to space limitation, we cannot present it using the full ODD protocol [7], but we keep only the main parts of it.

3.1 Overview of the Model

The main purpose of this approach is to model the disease spread in a network of cities using a coupled model composed of an aggregated equation-based approach (SIR) for the epidemic dynamic within cities and an agent-based approach for the epidemic spread between cities (air traffic). The two main processes we take into account are the local spread of the epidemic in each city and the spread through the network (we consider here only the air traffic).

3.2 Entities, State Variables and Scales

The model is composed of two kinds of agents: the *cities* (that are the *nodes* of the network) and the *airplanes* (that will carry passenger from one city to another one).

A city agent (defined in the *node* breed) is dedicated to describe the epidemic evolution of the whole population of a city. As this evolution will be described using the SIR model, the population of the city is represented by three state variables (S_Node, I_Node, R_Node) representing respectively the number of inhabitants in the Susceptible, Infected and Recovered states. It is important to notice that these state variables will contain float values (due to the integration of the SIR system using the Runge-Kutta 4 method). A city agent is also characterized by a mobility rate (*mobility-rate-node*) that will be used to compute the number of inhabitants that will travel. The number of people that should travel is stored in the variable *stock-to-flight*. To deal with mitigation strategies, two additional state variables are introduced (*in-airport* and *out-airport*): they take boolean values and represent the fact that the city accepts (or not) incoming flights and emits (or not) flights.

The airplane agents (defined in the *MobileGroup* breed) are characterized by their *speed* (that will be calibrated given the chosen disease and the spacial scale), their target city (*Next-Node*) and their population (*i.e.* the number of susceptible *S_Group*, infected *I_Group* and recovered *R_Group* people in the plane).

An edge of the network is only characterized by the two cities it connects. We consider it represents an air route without stopover. As a consequence, in this simple version of the model, we do not consider stopovers in the trips.

In order to allow the numerical integration of the SIR systems, we chose that the simulation step corresponds to the integration step. We thus need to synchronize the air traffic process on this time discretisation. The time and space scales are defined as parameters of the simulation. Given a disease (characterized by its *alpha* and *beta* parameters) and an initial population, the modeller can choose the size of the environment (*TerritorySize-km* in km) and the integration step (h). Then a calibration phase is done at the initialization to compute other parameters (*e.g.* speed of airplanes).

3.3 Process Overview and Scheduling

The 2 main processes in the model are (i) the epidemic evolution and (ii) the air traffic. In terms of scheduling, at each simulation step, first the model computes the disease spread in each city and each airplane and then each city computes whether it should create a new airplane and chooses its target. Finally each airplane moves toward its target and when it reaches it, its population is integrated into the city population.

Epidemic Evolution. In each city and airplane, the epidemic dynamics is driven by a SIR ODE system (*c.f.* Fig. 1). We consider only one disease and we do the hypothesis that the same system (with the same disease-related parameters) can be applied to each city and each airplane. So taken as input the number of people of a city in each state, we update these numbers by discretising the system using the Runge Kutta 4 numerical integration method[1]. This computes the new number of people in each state. It is important to notice that the number of people in each state computed using the chosen integration method is a float value.

[1] The numerical integration is done using an external plugin computing each RK4 integration step.

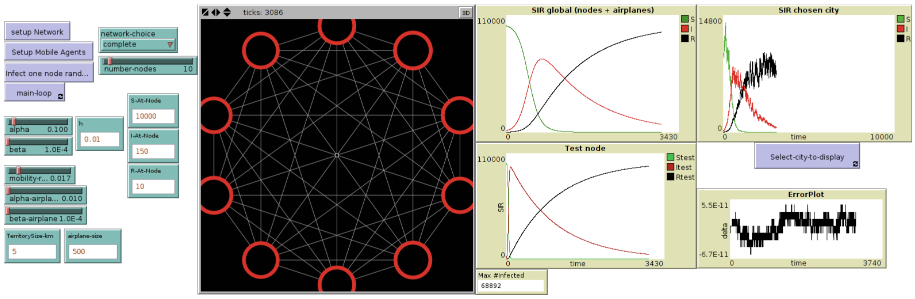
The equation-based approach has been chosen to describe the epidemic evolution in each city because this approach is dedicated to describe dynamics in a huge and homogeneous population, which is implicitly the case when we consider a city as one agent. In addition, it allows us to increase the population on each node without lose in terms of computation time.

Air Traffic. Each city has a variable *stock-to-flight* that represents the number of people that should leave the city and travel to another one. At each simulation step, each city computes (thanks to its *mobility-rate-node* attribute and its population) the number of people that should leave the city and add it to the *stock-to-flight* variable. If this number is greater than an airplane capacity, airplanes are created in order to empty this stock. Given the number of people that take an airplane, we extract a set of people that is representative of the whole city population by using a proportional random draw: for each individual to be added in the airplane we randomly choose his epidemic state. As a consequence, the number or Susceptible, Infected and Recovered people are initialized with an integer value: as the population in airplane is small we need to extract and manage individually each people.

Once an airplane has been populated, its target is randomly chosen among all the adjacent cities.

3.4 Initialization

Calibration. The duration of flights depends on the distance between the source and the target and is calibrated on the integration step. Therefore, the simulation setup contains a calibration step between the integration step, the flight duration and the distance. Given a particular disease (characterized by its *alpha*



network-choice: complete
number-nodes 10
alpha 0.100
beta 1.0E-4
h: 0.01
S-At-Node: 10000
I-At-Node: 150
R-At-Node: 10
mobility-r... 0.017
alpha-airpla... 0.010
beta-airplane 1.0E-4
TerritorySize-km: 5
airplane-size: 500
ticks: 3086
Max #Infected: 68892

Fig. 4. Interface of the MicMac model. The modeller can choose the topology of the network (among complete, random, small-world...), the mobility rate and the initial population. The interface allows the modeller to observe the epidemic dynamics of one city but also of the whole network. The difference between the current and the initial population (error) is computed at each step, in order to ensure that the population remains constant during the simulation (a divergence could appear as we both manipulate float and integer values for the population).

and *beta* (parameters) and its observed duration, the calibration is done on an additional reference node containing the population of the whole network: a SIR dynamics is applied until the epidemic end (*i.e.* until the step where the number of infected people is lower that an given threshold). This gives us the simulation step corresponding to the end of the epidemic and thus the duration corresponding to an integration step h. Given the size of the environment, we can thus adjust the flight speed.

As an example, if we consider a disease characterized by $\alpha = 0.2$, $\beta/N = 0.5$ and a duration of 60 days and an initial population of 39999 susceptible and 1 infected individuals in each of the 10 cities of the network, we get a step duration of 0.1 days. So 1 simulation step duration is about 2.4 h in this case, which justifies the need to take into account flights that are not instantaneous.

Parameters Values. Parameters of the MicMac model are based on the U.S. domestic flights data[2]. The 10 cities with the biggest airports have 57 709 474 inhabitants and about 21 420 000 passengers per month. The mobility rate is thus 0.37 per month and 0.012 per day. In addition, the area of the U.S. is about 9600000 km^2 and the mean size of airplanes is 80 passengers.

We chose thus parameters proportional to these values.

4 Evaluation Method

In order to assess the impact of the agent-based model coupled with SIR models, we will progressively study several heterogeneities introduced by the individual airplane transportations and compare them with a reference model. This model considers only the epidemic dynamic in the cities: it is thus the integration of several SIR nodes[3]. This model is also equivalent to a homogeneous meta-population model on a complete graph.

On this model we will consider three particular values (illustrated on the Fig. 5) as indicators:

- the maximum value of the number of infected people: MaxI;
- the time when the number of infected people is maximum: TimeofMaxI;
- the duration of the epidemic: Duration.

These three indicators will be denoted $MaxI_{nSIR}$, $TimeofMaxI_{nSIR}$ and $Duration_{nSIR}$ for the reference model and $MaxI_{MicMac}$, $TimeofMaxI_{MicMac}$ and $Duration_{MicMac}$ for the hybrid MicMac model.

Whereas the two last indicators are computed by (numerical) simulation, this model is interesting because the first one can be computed analytically. If we have n nodes with on each an initial population of I_{init} infected, S_{init} susceptible

[2] http://www.rita.dot.gov/bts/sites/rita.dot.gov.bts/files/press_releases/airline_traffic_data.html.

[3] This is not equivalent to 1 node containing the whole population.

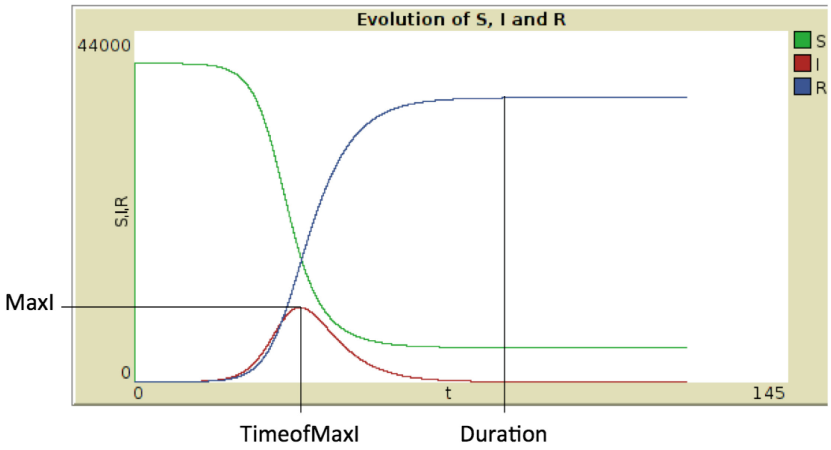


Fig. 5. Evolution of the number of susceptible, infected and recovered people over time, computed with $\alpha = 0.2$, $\beta/N = 0.5$ and $S_{init} = 1000$, $I_{init} = 10$, $R_{init} = 1$. The numerical integration method was Runge Kutta 4 with an integration step $h = 10^{-3}$.

and R_{init} recovered ($N = S_{init} + I_{init} + R_{init}$ being the total number of people), the value of MaxI is:

$$MaxI_{nSIR} = n\left(I_{init} + S_{init} + \frac{N\alpha}{\beta}\left(-1 + \ln\left(\frac{N\alpha}{\beta}\right) - \ln\left(S_{init}\right)\right)\right)$$

5 Comparison

From a mean field model fitting with the reference model (Sect. 5.1), we progressively introduce heterogeneity on various dimensions: (i) initial population distribution (Sect. 5.2), (ii) time of flights (Sect. 5.3) and (iii) network (Sect. 5.4) and evaluate their impact.

5.1 Reference Case: Equivalence of both Models

In this model, we remove all heterogeneity and the effect of the spatial component on the model:

- the mobility rates and initial population are the same for every node (no heterogeneity among cities).
- flights are instantaneous (the model is aspatial);
- the network is a complete network (no heterogeneity due to the network);

We compute (by simulation) the various indicators and we get the following results:

- $MaxI_{nSIR} = MaxI_{MicMac}$;

- $TimeofMaxI_{nSIR} = TimeofMaxI_{MicMac}$;
- $Duration_{nSIR} = Duration_{MicMac}$.

This first experiment shows that the MicMac model fits the reference model: we have thus built an agent-based model able to reproduce a set of equation-based models under mean field assumptions. This initial step was necessary to carefully assess the following models. From this we can go deeper in the exploration of the impact of heterogeneity in our agent-based model.

5.2 Impact of Initial Conditions (Populations)

Instead of having a homogeneous population in each node, we introduce heterogeneity among nodes by creating initially all the infected people in only one city (this represents a more realistic situation where an epidemic starts in one location before spreading). Nevertheless the global population remains the same in the two models. Other hypotheses made in the previous section remain true. We get the following results:

- $MaxI_{nSIR} > MaxI_{MicMac}$;
- $TimeofMaxI_{nSIR} < TimeofMaxI_{MicMac}$;
- $Duration_{nSIR} < Duration_{MicMac}$.

All these indicators show that we now have a disease spread behaving as a classical diffusion phenomenon: the maximum of the epidemic is lower in the MicMac model because all the cities are not synchronized anymore and the TimeofMaxI is postponed (which induces a longer epidemic). We have to notice that the network topology has no effect here (as it is complete), but the diffusion is not instantaneous because of the fact that an airplane without infected people can leave an infected city (due to the random filling of the airplanes).

5.3 Impact of the Time in Flight

From the MicMac model presented above (with initial heterogeneous population distribution), we now release the instantaneity of flights. The size of the area is now taken into account and thus travels now take several simulation steps: the temporal coupling between travel time and integration step is made at initialization during the calibration step (*c.f.* Sect. 3.4).

We have explored the impact of the size of the area over the three indicators and the results are presented in Fig. 6. We observe that the size of the area does not have a significant impact on the results (for all the indicators).

This result is quite surprising as it shows that the distance has no influence on results in the current MicMac model. We can imagine two explanations:

- the total flying population is constant (*i.e.* independent on the travel distance and time);
- the contagion model in the airplanes is not adapted (the population in airplanes is small, so out of the scope of the classical SIR model).

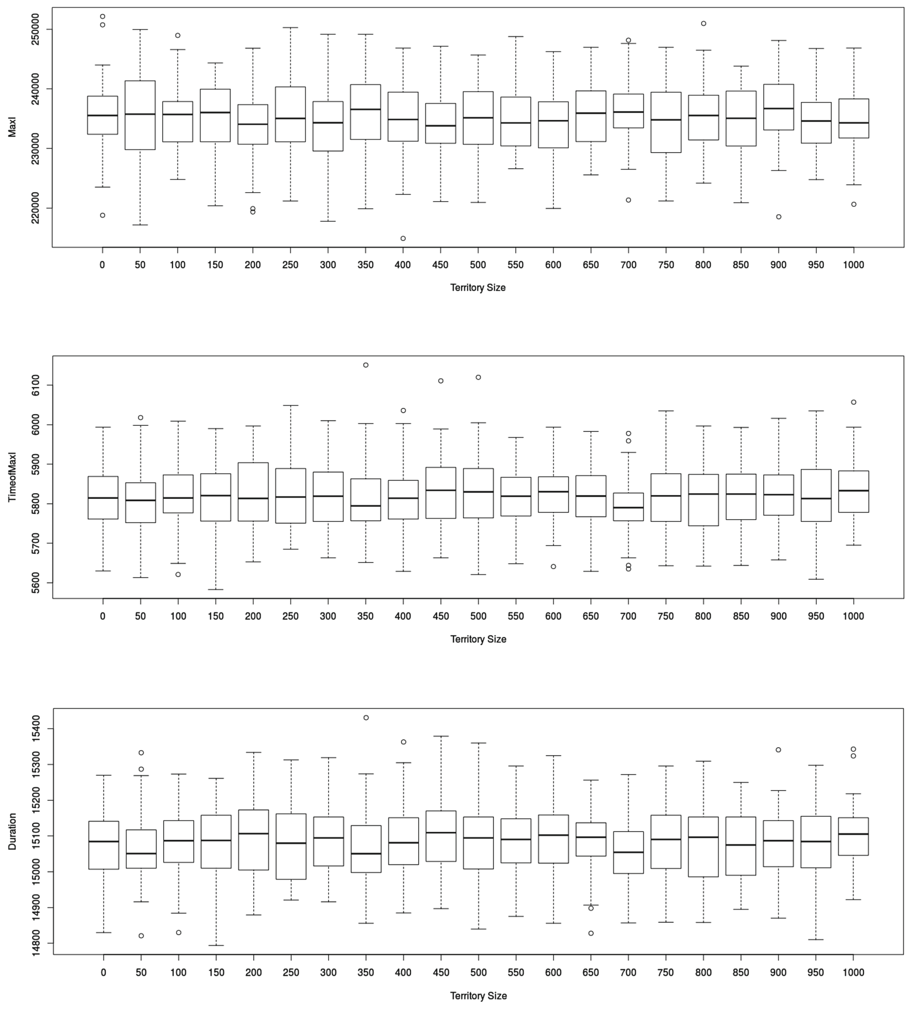


Fig. 6. Influence of the area size on the indicators MaxI, TimeofMaxI and Duration

It could be interesting to investigate the second explanation. A way to do could be to replace the ODE model by a Gillespie algorithm that is dedicated to small population (everything else in the model remaining unchanged).

5.4 Impact of the Network

In order to study the impact of the network topology on results, we consider the same model initialized with various networks. In every case, we consider a network with 100 nodes (cities). First we use regular networks with 4 to 99 neighbours for each node (the last one is a complete network). Then we use a small-word network. To produce small-world networks, we use the Watts &

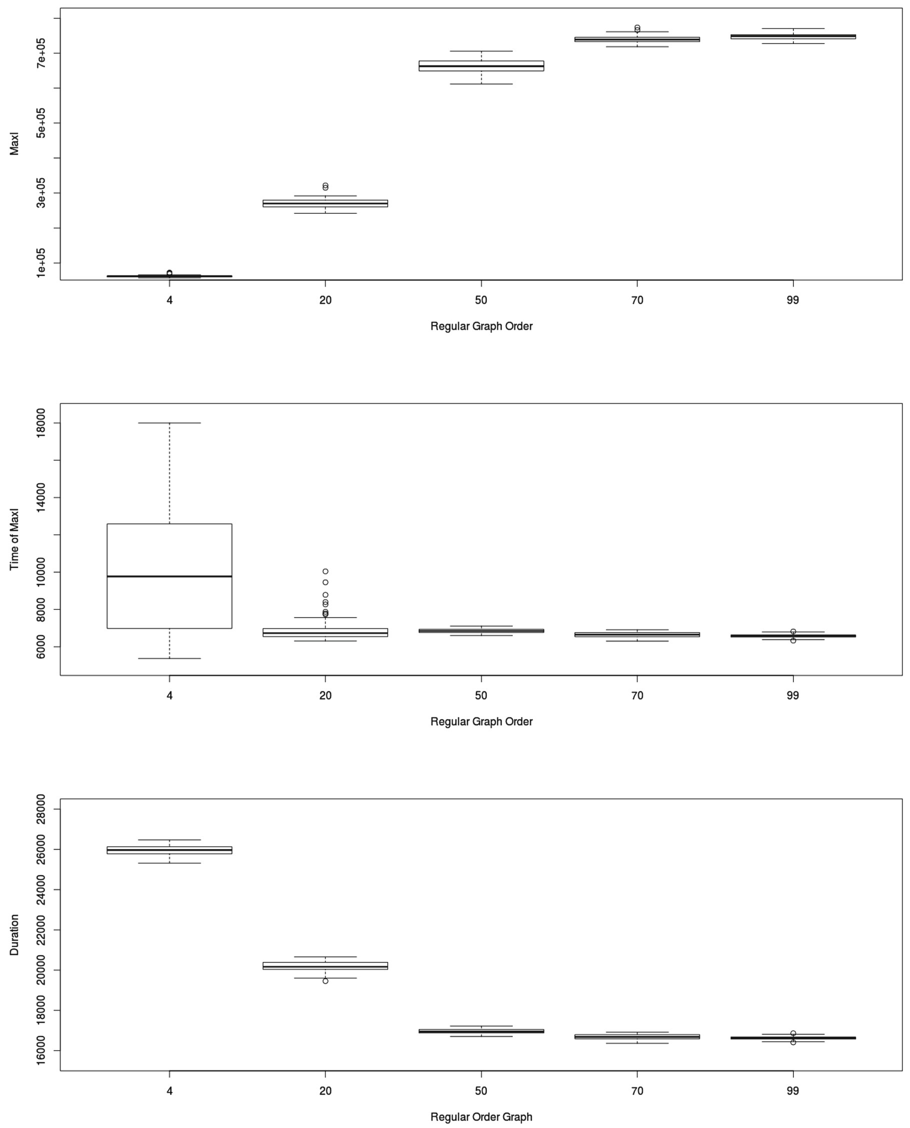


Fig. 7. Indicators values for regular networks with order among {0.04, 0.2, 0.5, 0.7, 0.99}

Strogatz algorithm [15]: we start with a regular network with 4 neighbors for each node. Then we rewire some of the nodes. To evaluate the impact of various small-world network, we consider rewiring probabilities from 0.2 to 1. Results are summarized in Figs. 7 and 8.

In the case of regular networks, we can observe that the MaxI value increases and the TimeofMaxI and Duration decrease when the number of edges increases.

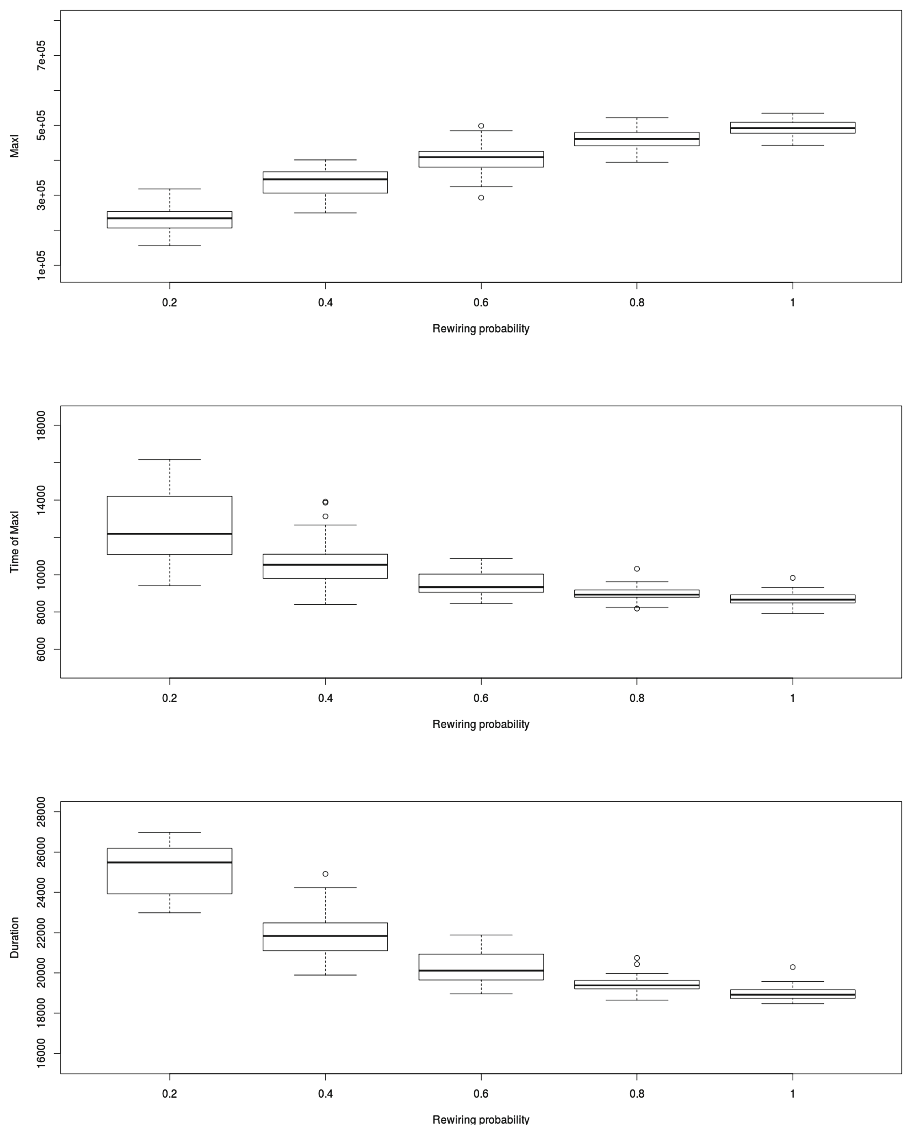


Fig. 8. Indicators values for small-world networks with rewiring probability among {0.2, 0.4, 0.6, 0.8, 1}

It is due to the fact that it is easier to infect other nodes when it is easier to access them via the network (which is allowed by increasing the number of edges). In the case of the small-world networks, we have similar observation when we increase the rewiring probability. The rewiring indeed creates shortcut in the network. The more shortcuts the network has, the easier it is to infect other nodes.

Rewiring probability	0.2	0.4	0.6	0.8	1
APL	4.23	3.73	3.46	3.44	3.41

Fig. 9. Average path length for various small-worlds networks

Order	0.04	0.2	0.5	0.7	0.99
APL	12.88	2.98	1.5	1.22	1

Fig. 10. Average path length for various regular networks

In fact, increasing the rewiring probability (small-world networks) or the network order (regular networks) have the same effect: they decrease the diameter and the average path length (APL) of the network (*c.f.* Figs. 9 and 10). In addition it is interesting to notice that the topology itself has not such an influence on the result, mainly the diameter (and the APL) of the network has the higher influence on results[4].

6 Perspectives: Application of Mitigation Strategies

Once the model being designed, we can use it to evaluate several mitigation strategies. Due to space limitation, we give here only the method to implement these strategies. Several strategies have been proposed in the literature [3,11], such as the quarantine, avoidance of risk culture (*c.f.* Sect. 2.2).

The two first strategies are parameterized by a threshold: a city is put in quarantine or avoidance when its rate of infected people is greater than the given threshold. As an example, in the quarantine strategy, we consider that a city is put in quarantine when the ratio of infected people reached the threshold $\theta_{quarantine}$, *i.e.* when the following condition is fulfilled:

$$\frac{I_i}{S_i + I_i + R_i} > \theta_{quarantine}$$

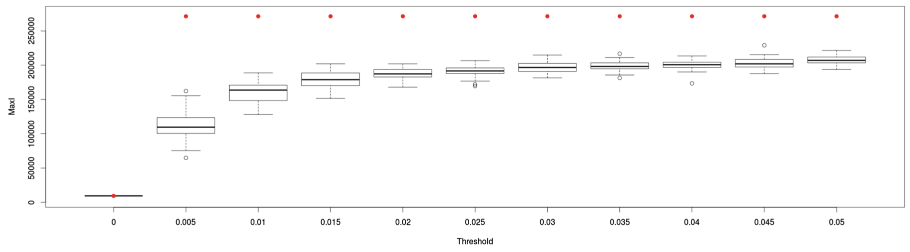


Fig. 11. Influence of the quarantine threshold on the indicator MaxI for the MetaPop (red plots) and the MicMac models (Color figure online).

[4] This is only a first attempt of result: it should be verified on other network topologies. Other characteristics should also be tested.

We might consider that such strategy of containment is well defined at a macroscopic level and directly compatible with the metapopulational model. Therefore, adding a microscopic component would not add much. However, simulations run with both models suggest a different conclusion (Fig. 11). While the dynamics of the epidemic is largely driven by the SIR dynamics on nodes composed of large static populations (compared to small flying populations), the discretisation and stochasticity implied by the agent formalism has a significant impact on containment strategies, even if they rely to a macroscopic level. Indeed, as can be seen on Fig. 11, the impact of such a quarantine strategy on the epidemic, defined by the three indicators MaxI, TimeofMaxI and Duration, is much more progressive and realistic.

7 Conclusion

We have presented the MicMac model that is a model coupling equation-based and agent-based models in a model representing the spread of a disease in a city network. We have studied carefully the influence of heterogeneities due to the agent-based model and compared results *w.r.t.* a reference model that does not contain the agent part. Finally we have presented how to integrate mitigation strategies inside.

As perspectives we plan to show the benefits of the hybrid approach (and in particular the benefits brought by the agent approach) by introducing much more complex diffusion processes: in particular, passengers will be able to do multistop travels, with possibility to be infected in the transit airport and passengers will be able to do round trips (they will be able to bring back disease to their home city).

Acknowledgements. This work has been partially founded by the CNRS through the PEPS HuMaIn (2013 and 2014) and by the French National Network of Complex Systems (RNSC) through the interdisciplinary network MAPS (http://maps.csregistry.org/).

References

1. Arino, J., Van den Driessche, P.: A multi-city epidemic model. Math. Popul. Stud. **10**(3), 175–193 (2003)
2. Cecconi, F., Campenni, M., Andrighetto, G., Conte, R.: What do agent-based and equation-based modelling tell us about social conventions: the clash between abm and ebm in a congestion game framework. J. Artif. Soc. Soc. Simul. **13**(1), 6 (2010)
3. Colizza, V., Vespignani, A.: Epidemic modeling in metapopulation systems with heterogeneous coupling pattern: theory and simulations. J. Theor. Biol. **251**, 450–467 (2008)
4. Durrett, R., Levin, S.: The importance of being discrete (and spatial). Theor. Popul. Biol. **46**(32), 363–394 (1994)

5. Fahse, L., Wissel, C., Grimm, V.: Reconciling classical and IB approaches in theoretical population ecology: a protocol for extracting population parameters from IBM. Am. Nat. **152**, 838–852 (1998)
6. Forrester, J.W.: Industrial Dynamics. MIT Press, Cambridge (1961)
7. Grimm, V., Berger, U., DeAngelis, D.L., Polhill, J.G., Giske, J., Railsback, S.F.: The ODD protocol: a review and first update. Ecol. Model. **221**(23), 2760–2768 (2010)
8. Hanski, I.A., Gilpin, M.E. (eds.): Metapopulation Biology: Ecology, Genetics, and Evolution. Academic Press, Waltham (1997)
9. Kermack, W.O., McKendrick, A.G.: Contributions to the mathematical theory of epidemics. J. Hyg. **39**(3), 271–288 (1939)
10. Laperrière, V., Badariotti, D., Banos, A., Müller, J.-P.: Structural validation of an individual-based model for plague epidemics simulation. Ecol. Complex. **6**(2), 102–112 (2009)
11. Meloni, S., Perra, N., Arenas, A., Gomez, S., Moreno, Y., Vespignani, A.: Modeling human mobility responses to the large-scale spreading of infectious diseases. Sci. Rep. **1**, **62**(7), 1–7 (2011)
12. Van Dyke Parunak, H., Savit, R., Riolo, R.L.: Agent-based modeling vs. equation-based modeling: a case study and users' guide. In: Sichman, J.S., Conte, R., Gilbert, N. (eds.) MABS 1998. LNCS (LNAI), vol. 1534, pp. 10–25. Springer, Heidelberg (1998)
13. Press, W.H., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical Recipes. The Art of Scientific Computing, 3rd edn. Cambridge University Press, Cambridge (2007)
14. Simoes, J.: An Agent-Based Approach to Spatial Epidemics through GIS. Ph.D. thesis, University College London (2006)
15. Watts, D.J., Strogatz, S.H.: Collective dynamics of 'small-world' networks. Nature **393**(6684), 440–442 (1998)
16. Wilensky, U., Evanston, I.: Netlogo. Center for connected learning and computer based modeling. Technical report, Northwestern University (1999)

Validation and Verification in MABS

MC^{2}MABS: A Monte Carlo Model Checker for Multiagent-Based Simulations

Benjamin Herd(✉), Simon Miles, Peter McBurney, and Michael Luck

Department of Informatics, King's College London, London, UK
benjamin.c.herd@kcl.ac.uk

Abstract. Agent-based simulation has shown great success for the study of complex adaptive systems and could in many areas show advantages over traditional analytical methods. Due to their internal complexity, however, agent-based simulations are notoriously difficult to verify and validate.

This paper presents MC^{2}MABS, a Monte Carlo Model Checker for Multiagent-Based Simulations. It incorporates the idea of *statistical runtime verification*, a combination of statistical model checking and runtime verification, and is tailored to the approximate verification of complex agent-based simulations. We provide a description of the underlying theory together with design decisions, an architectural overview, and implementation details. The performance of MC^{2}MABS in terms of both runtime consumption and memory allocation is evaluated against a set of example properties.

Keywords: Agent-based simulation · Verification · Formal methods · Testing

1 Introduction

Agent-based simulation (ABS) is rapidly emerging as a popular paradigm for the simulation of complex systems that exhibit a significant amount of non-linear and emergent behaviour [28]. It uses populations of interacting, heterogeneous and often adaptive agents to model and simulate various phenomena that arise from the dynamics of the underlying complex systems. Although social science has been its traditional domain, ABS is also increasingly being used for the analysis of complex (socio-)technical, often also safety-critical systems in areas such as avionics [5], the design and analysis of robot and UAV swarms [30], and increasingly also the Internet of Things [17].

In that context, and similar to other software systems, *correctness* plays a central role and questions of quality assurance become increasingly important. It is common to distinguish between *verification* and *validation*. Whereas the former is targeted towards a system's correctness with respect to its specification (i.e. its correct implementation), the latter ensures a sufficient level of accuracy

© Springer International Publishing Switzerland 2016

with respect to the intended application domain. In the context of modelling and simulation, the distinction between verification and validation becomes more complicated; it can thus be more useful to talk about *internal* and *external validation* instead [4,34].

With their ability to produce complex, emergent behaviour from the action and interaction of its components, ABSs are notoriously difficult to understand and to engineer. In the multiagent systems community, formal and semi-formal verification approaches (both qualitative and quantitative) have shown great power. Methods and tools for the verification of ABS in particular, however, are still largely missing. With potentially large populations, different observational levels, heterogeneity, a strong focus on emergent behaviours, and a significant amount of randomness, ABSs possess a range of characteristics which, in their combination, represent a great challenge for verification.

In this paper, we present our research efforts on the ABS verification problem and introduce `MC`^{2}`MABS`— the *Monte-Carlo Model Checker for Multiagent-Based Simulations* — a prototypical statistical runtime verification approach and framework for complex agent-based simulation models. We start with a motivational example from the area of swarm robotics in Sect. 2. An overview of related work and some theoretical background on statistical model checking and runtime verification is given in Sects. 3 and 4. The framework itself is described in Sect. 5, followed by a brief performance evaluation w.r.t. both runtime and memory consumption in Sect. 5.3. The paper concludes with a summary and ideas for future work.

2 Motivational Example: Collective Behaviour in Swarm Robotics

In order to motivate the usefulness of verification for the analysis of agent-based simulation models, we introduce a small scenario from the area of swarm robotics. Although purely formal approaches for the analysis of swarm robotic models have shown to be useful [19,23], they are not always applicable. For example, in order to be analytically tractable, purely formal approaches typically pose strong homogeneity assumptions upon the individual agents. This is appropriate as long as details of the environment, particular interactions, and individual differences (e.g. w.r.t. faulty behaviour) are irrelevant for the analysis. However, many emergent phenomena only become apparent if interaction, heterogeneity, and locality are taken into account. In this case, formal analysis may become intractable and the only way to investigate the dynamics of the scenario is simulation.

We focus here on *swarm foraging*, a problem which has been widely discussed in the literature on cooperative robotics [7]. Foraging describes the process of a group of robots searching for food items, each of which delivers energy. Individual robots strive to minimise their energy consumption whilst searching in order to maximise the overall energy intake. The study of foraging is important because it represents a general metaphor to describe a broad range of (often critical) collaborative tasks such as *waste retrieval*, *harvesting* or *search-and-rescue*.

A good overview of multirobot foraging has been given by Cao *et al.* [7]. In a foraging scenario, robots move through the space and search for food items. Once an item has been detected within the robot's field of vision, it is brought back to the nest and deposited which delivers a certain amount of energy to the robot. Each action that the robot performs also consumes a certain amount of energy. The overall swarm energy is the sum of the individual energy levels.

A designer's main challenge is to tune the parameters of the individual agents such that the swarm as a whole is able to *self-organise* efficiently, i.e. to adapt to environmental circumstances such as food density. Since there is no central control, adaptation has to *emerge* from the agents' local actions and interactions. Due to the irreducibility of emergent phenomena, designing a distributed algorithm with a particular emergent behaviour in mind can be highly non-trivial. Interesting mechanisms have been presented by Liu *et al.* [24]. Here, agents adapt their behaviour according to three *cues*: (i) internal cues (personal success in retrieving food), (ii) environmental cues (collisions with teammates while searching for food), and (iii) social cues (teammate success in food retrieval). Depending on those influences, agents increase or decrease their searching and resting times with the goal of achieving an optimal collective division of labour.

ABS represents a powerful approach to study the dynamics of a distributed swarm algorithm. However, tuning the individual parameters such that the overall emergent behaviour is optimal can be hard. Verification — both qualitative and quantitative — can be of great help during the design process. As a starting point, one could, for example, formulate and verify the following qualitative safety property upon the simulation model: "the swarm must never run out of energy" (formally: $\neg\ \mathbf{F}(\mathit{energy} \leq 0)$). Although useful, such a pure macro-level criterion is rarely sufficient since, despite the whole swarm always having enough energy, individual agents may still run out. In order to solve this problem, a more fine-grained, quantified criterion may be formulated. Rather than stating that the swarm as a whole must never run out of energy, one may, for example, stipulate that "no individual agent must ever run out of energy" (formally: $\forall\ a \bullet \neg\ \mathbf{F}(\mathit{energy}_a \leq 0)$). Checking this criterion would catch those cases in which individual agents run out of energy, but one would still not know (i) *how many* of them do, and (ii) *why* this is the case.

In addition to conventional qualitative safety and liveness checking which provides clear yes/no answers but little explanatory insights, *quantitative analysis* may help to shed further light on the dynamics of the system. In addition to the two safety criteria above, it may, for example, be useful to answer the following questions.

- What is the *avg./min./max. probability* of an agent running out of energy?
- What *fraction of time* does an agent spend homing/resting/etc.?
- What *fraction of time* does an agent spend transitioning, e.g. from depositing to resting? (= overall probability of recharging)
- *How likely* is an agent to transition
  - from searching to grabbing? (= probability of finding food)
  - from grabbing to depositing? (= probability of losing out on food)

– What is the *correlation* between an agent's type and its probability of doing sth.?

Summarising the example above, we believe that a verification approach for ABS needs to satisfy the following requirements.

Efficiency: the approach should allow for the verification of complex simulation models in a timely manner. Due to the highly iterative nature of the modelling process, a user should be able to choose between a high level of accuracy at the expense of verification time and a lower level of accuracy but quicker results.

Expressivity: properties need to be formulable in a *formal, unambiguous way* and verifiable on *different levels of observation*: individual agents, groups of agents, as well as the whole population. Furthermore, the approach should allow for the verification of *both qualitative and quantitative properties*.

Flexibility: due to the sensitivity of complex systems to local differences and environmental conditions, a verification approach should not impose any unrealistically strict limitations on the simulation models that are verifiable, e.g. by assuming that agents are entirely homogeneous, by abstracting away the environment, etc.

Immediacy: in order to ensure the relevance of the verification results, the gap between the model to be verified and the actual simulation model should be as small as possible. Ideally, verification is performed *upon the original simulation model*.

As described in the following section, existing approaches do not currently satisfy those requirements in their combination. With the framework presented in this work, we aim to close this gap.

3 Related Work

Model Checking Multiagent Systems: Since its beginnings around 30 years ago, model checking has gained huge significance in computer science and software engineering in particular and has been successfully applied to many real-world problems. Model checking has also gained increasing importance in the multiagent community and numerous approaches have been presented in literature [9]. In alignment with the classical problems studied in the community, multiagent verification typically focusses on qualitative properties involving notions such as time, knowledge, strategic abilities, permissions, obligations, etc. In order to allow for the verification of larger agent populations, model checking algorithms for temporal-epistemic properties have also been combined successfully with ideas such as bounded model checking [26], partial order reduction [25] and parallelisation [20]. Despite impressive advances, however, verification still remains limited to either relatively small populations or scenarios with strong homogeneity assumptions [33].

In recent years, probabilistic approaches to model checking have also gained increasing importance in the multiagent community. Examples include the verification of systems with uncertainty w.r.t. communication channels and actions [11], qualitative and quantitative analysis of agent populations with uncertain knowledge [37], verification of probabilistic swarm models [18], or automated game analysis [3]. Similar to their non-probabilistic counterparts, these approaches also suffer from the state space explosion and are thus either limited to relatively small systems or dependent upon strong homogeneity or symmetry assumptions which increase their scalability but also limit their applicability to the verification of complex simulation models.

Verification of Agent-Based Simulations: In the simulation community, most work on quality assurance focusses on validation, in particular statistical analysis of simulation output [29,35]. Albeit related, those approaches possess a different flavour than the one described in this work since they focus on the *external validity* of the model, i.e. the link between the model world and the real world[1]. Verification (or *internal validation*), on the other hand, focusses on the link between the model and the theory. If mentioned at all in the simulation literature, verification is mostly equated with conventional code verification, e.g. through testing, reviews, or static analysis.

In the ABS community, a number of testing and monitoring approaches have been presented [6,32,39]. Most of these approaches are based on top of existing modelling frameworks such as Repast or Mason; consequently, correctness properties (in the form of test cases) are formulated in the target language, typically in Java. An approach that combines ABS with numeric analysis has been presented by Wolf *et al.* [10]. The main motivation of the work is to determine if an individual-based system exhibits certain macroscopic emergent behaviour. To that end, repeated simulation is paired with numeric analysis from the system dynamics domain in order to detect deviations or to approximate the steady-state behaviour of the simulation. An advantage of the approach is its ability to speed up simulation by *steering* it into the direction required by the analysis algorithm. Due to its global focus, the approach is restricted to macro-level analysis; nevertheless, the power lies in the fact that it allows for the analysis of properties which conventional testing is not able to deal with.

Semi-formal and formal verification approaches similar to those for MAS described above but particularly tailored to ABS are still largely missing. The work closest related to ours is that of Sebastio and Vandin [36]. They present MultiVeStA, a statistical analysis tool which pairs discrete event simulation with statistical model checking. Similar to MC^{2}MABS, MultiVeStA can be coupled with existing simulators and allows for the verification of properties about expected values of observations. Properties are propositional in nature, i.e. more complex calculations or aggregations have to be 'wrapped' into propositions. Since MultiVeStA is not tailored to ABS, no internal structure is imposed on the simulation traces. As a consequence, there is no direct notion of observational levels in the property specification language.

[1] McKelvey refers to this link as the model's *ontological adequacy* [31].

It is important to note that, although useful in its own right, (semi-)formal verification will be even more powerful if it is embedded in a proper experimental environment. It has been argued that, if the systematic design of experiments was fully realised, the transparency of a simulation model could be increased significantly [27]. We believe that, rather than serving as an alternative, (semi-) formal verification may well become an integral part of this process.

4 Background

Statistical Model Checking: Conventional model checking aims to find an accurate solution to a given property by exhaustively searching the underlying state space which is, in general, only possible if the space is of manageable size [2]. One solution that works for probabilistic systems is to use a *sampling approach* and employ statistical techniques in order to generalise the results to the overall state space. In this case, n paths or *traces* are sampled from the underlying state space and the property is checked on each of them; statistical inference, e.g. *hypothesis testing*, can then be used to determine the significance of the results. Approaches of this kind are summarised under the umbrella of *statistical model checking*; a good overview is given by Legay *et al.* [21]. Due to its independence of the underlying state space, statistical model checking allows for the verification of large-scale systems in a timely, yet approximate manner.

One particular approach, *Approximate Probabilistic Model Checking*, provides a *probabilistic guarantee* on the accuracy of the approximate value generated by using *Hoeffding bounds* on the tail of the underlying distribution [12]. According to this idea, $\ln\left(\frac{2}{\delta}\right)/2\epsilon^2$ samples need to be obtained in order for the estimated probability Y to deviate from the real probability X by at most ϵ with probability $1-\delta$, i.e. $Pr(|X-Y| \leq \epsilon) \geq 1-\delta$. In this case, the number of traces grows logarithmically with δ and quadratically with ϵ. It is, however, interesting to note that the sample size is completely independent from the size of the underlying system.

Runtime Verification: Runtime verification attempts to circumvent the combinatorial problems of conventional model checking by focussing on the *execution trace* of a system rather than on its universal behaviour and performing correctness checks on-the-fly [22]. Due to its focus on the *execution trace* of a running system, it avoids most of the complexity problems that are inherent to static techniques; in that respect, runtime verification bears a strong similarity to testing. However, in contrast to conventional testing, runtime verification typically allows for the formulation of the system's desired behaviour in a more rigorous way, e.g. using temporal logic, and can thus be considered more formal. In general, runtime verification provides a nice balance between rigorous and strong but complex formal verification one hand, and efficient but significantly weaker conventional software testing on the other hand. It can thus be seen as a lightweight alternative for systems that are not amenable to formal verification.

Due to its focus on individual execution traces, runtime verification views time as a linear flow and properties are thus often formulated in a variant of linear temporal logic (LTL). Those properties are then translated into a *monitor* which is used to observe the execution of the system and report any satisfaction or violation that may occur. In order for a monitoring approach to be efficient, it necessarily needs to be *forward-oriented*; having to rewind the execution of a system in order to determine the truth of a property is generally not an option. In terms of monitor construction, two different approaches, *automaton-based* and *symbolic*, can be distinguished. In this work, we follow a symbolic approach which strongly relies upon the notion of *expansion laws.* Informally, expansion laws allow for the decomposition of an LTL formulae into two parts: the fragment of the formula that needs to hold in the *current* state and the fragment that needs to hold in the *next* state in order for the whole formula to be true. It is useful to view both fragments as *obligations*, i.e. aspects of the formula that the trace under consideration needs to satisfy immediately and those that it promises to satisfy in the next step. For example, the expansion law for the 'until' operator of LTL is shown below:

$$\phi_1 \, \mathbf{U} \, \phi_2 \equiv \phi_2 \vee (\phi_1 \wedge \mathbf{X}(\phi_1 \, \mathbf{U} \, \phi_2)) \tag{1}$$

The equivalence states that, in order for a formula $\phi_1 \, \mathbf{U} \, \phi_2$ to be satisfied at time t, either (i) ϕ_2 needs to be satisfied at time t, or (ii) ϕ_1 needs to be satisfied at time t and $\phi_1 \, \mathbf{U} \, \phi_2$ needs to be satisfied at time $t + 1$. Expansion laws play an important role for the idea of runtime verification since they form the basis for a decision procedure which can be used to decide in a certain state if a given property has already been satisfied or violated. By decomposing a formula into an immediate and a future obligation, optimality can be achieved: as soon as the immediate obligation is satisfied and no future obligation has been created, the entire formula is satisfied and the evaluation finishes.

5 The Verification Framework

Considering the complexity of ABSs, we believe that a combination of statistical model checking and runtime verification, which we refer to as *statistical runtime verification (SRV)*, may serve as an interesting alternative to formal verification. Due to their probabilistic nature, ABSs can be seen as special variants of Monte Carlo simulations and each execution thus naturally produces a random trace of the underlying space. By verifying a property on a sufficiently large number of simulation runs, its probability can thus be estimated to an arbitrary level of precision. Clearly, the usefulness of this idea is critically dependent on the number of traces analysed. As described further below, the efficiency of each trace check can be improved significantly by interleaving simulation and verification. As opposed to formal macro-level analysis, SRV preserves the individual richness of the ABS and allows for the verification of interesting properties in a semi-formal way. Due to its focus on independent traces, SRV is easily parallelisable and thus highly scalable by exploiting the power of modern parallel hardware.

MC²MABS is a practical framework that incorporates the idea of SRV. Its design is based on four central requirements, as informally motivated in Sect. 2:

(i) *efficiency* (timely and tunably accurate verification of large-scale ABSs), (ii) *expressivity* (formulation and verification of qualitative and quantitative correctness properties in a formal, rigorous way), (iii) *flexibility* (verification of arbitrary ABSs), and (iv) *immediacy* (verification of the ABS itself, not a simplified model thereof). An overview of the framework is given below, the source code as well as additional documentation is available online [1].

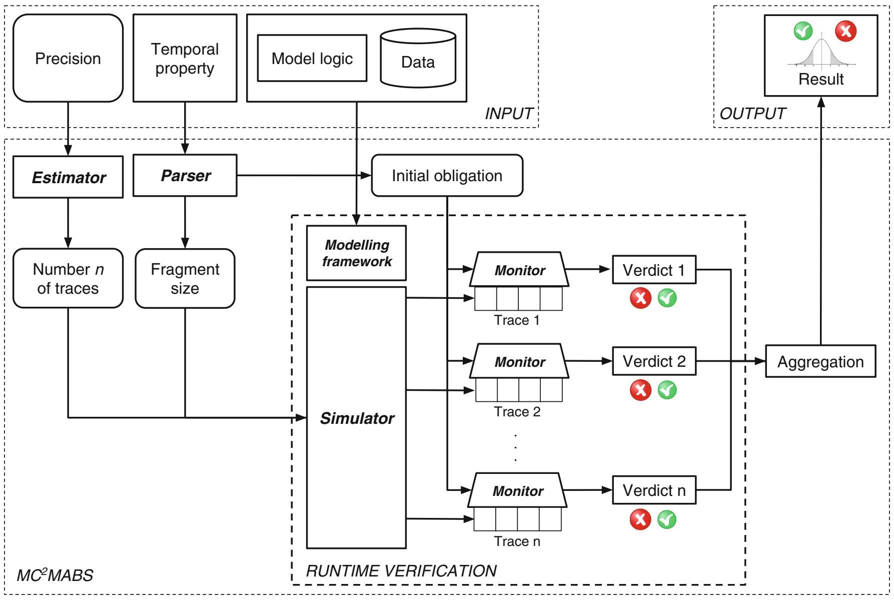


Fig. 1. The overall architecture of MC^{2}MABS

5.1 Architectural Overview

A high-level overview of MC^{2}MABS is shown in Fig. 1. The framework comprises as its central components (i) an estimator, (ii) a modelling framework, (iii) a property parser, (iv) a simulator, and (v) a runtime monitor. All components are described in more detail in Sect. 5.2 below. The typical sequence of actions in a verification experiment using MC^{2}MABS is as follows:

1. The user provides (i) the logic of the ABS by utilising the modelling framework, (ii) an associated correctness property, and (iii) the desired precision of the verification results as inputs to the verification framework.
2. The correctness property is parsed by the property parser and transformed into an expanded property that is used by the runtime monitor.
3. The estimator determines the number of simulation traces necessary to achieve the desired level of precision.

4. The simulator uses the model together with additional configuration information to produce a set of simulation traces.
5. Each simulation trace is observed by a runtime monitor which assesses the correctness of the trace using a given correctness property; due to the online nature of the monitor, a verdict is produced as soon as possible.
6. The individual results are aggregated into an overall verification result and presented to the user.

Due to the decoupling of simulation and verification, MC^{2}MABS supports both *ad-hoc* and *a-posteriori* verification. Ad-hoc verification is synonymous to runtime verification and assesses the correctness of a system during its execution. A-posteriori verification assumes the existence of traces prior to the actual verification. The latter mode can be useful, for example, if the traces have been obtained with a different simulation tool, e.g. NetLogo [38] or Repast [8]. In that case, the simulator of MC^{2}MABS is merely used to 'replay' the pre-existing output for the purpose of verification.

5.2 Components

Estimator: The main purpose of the estimator is to determine the number of traces necessary to achieve a certain level of precision (provided by the user) w.r.t. the verification results. MC^{2}MABS uses an algorithmic variant of the Hoeffding bounds briefly mentioned in Sect. 4. Due to its approximate nature, the Hoeffding bound often overestimates the actually required number of samples by a significant degree. The procedure we use instead operates directly on the Binomial distribution [13]. It has the same theoretical dependence on δ and ϵ but, due to its accurate nature, it returns a lower *total sample size.* In the presence of resource constraints, this (theoretically irrelevant) difference can represent a critical practical advantage. Different levels of precision and their corresponding sample size are shown below:

1. confidence $\delta = 99\,\%$, accuracy $\epsilon = 1\,\% \Rightarrow$ 13,700 traces
2. confidence $\delta = 99.9\,\%$, accuracy $\epsilon = 0.1\,\% \Rightarrow$ 24,000 traces
3. confidence $\delta = 99.9$, accuracy $\epsilon = 0.1\,\% \Rightarrow$ 2,389,000 traces

Modelling Framework: Instead of providing a dedicated model description language — a path taken by most existing verification tools — we decided to allow for the formulation of the underlying model in a high-level programming language. This is motivated by the observation that ABSs often contain a significant level of functional complexity (probability evaluations, loading and manipulation of external data, location-based search algorithms, etc.). Any simple modelling language would thus significantly (and unnecessarily) limit the range of models which it is capable of describing. As a consequence, we decided to take a different path and realise the communication between the simulation model and the monitor through a *service provider interface (SPI)* which provides a basic skeleton for the underlying model and limits the prescriptive part of the framework to a handful of callback functions. In order to maintain a high level of performance (which is crucial for the generation of large batches of traces), we use C++ as the modelling language. As a compiled multi-paradigm language, C++ offers a good balance between usability and performance.

Alternatively, rather than hosting the actual model logic, the modelling framework can also be used to implement logic that controls an external simulation tool such as NetLogo or Repast (e.g. running in 'headless mode'), collects the resulting data and forwards it to the verifier. In this case, $\texttt{MC}^2\texttt{MABS}$ acts as a 'man-in-the-middle' and extends existing simulation frameworks with a verification capability.

Property Parser: The property parser is responsible for translating a textual representation of a correctness property into an expanded version which is then used by the monitor to observe the temporal dynamics of a simulation trace. The parser uses a formal grammar that defines the space of valid properties. $\texttt{MC}^2\texttt{MABS}$ supports *simLTL*, a variant of LTL tailored to the formulation of properties about ABS traces [14]. As opposed to conventional LTL, simLTL allows for the formulation of properties about *individual agents* as well as about arbitrary *groups of agents*. This is achieved by a subdivision of the language into an *agent layer* and a *global layer*. Furthermore, the language is augmented with *quantification* and *selection* operators. These features make it possible to formulate properties such as the following:

- It is true for *every* agent that the energy level will never fall below 0
- No more than *20 %* of the agents will eventually run out of energy
- Agents *of group x* are *more likely* to run out of energy than those of *group y*

Furthermore, as explained below, the formulation of properties is closely linked with the way the simulator performs the sampling from the probability space underlying the simulation model. This is also accommodated by the property specification language which allows for (i) the annotation of properties in order to denote the length of *trace fragments* required for their verification as well as (ii) the formulation of higher-order properties, e.g. about the correlation of events, as described in the next paragraph.

Simulator: The simulator is responsible for executing the simulation model repeatedly in order to obtain a set of traces used for subsequent verification by the monitor. Technically, by repeatedly executing the simulation model, the simulator performs a sampling from the underlying probability space. By interpreting the probability space in different ways, different levels of granularity with respect to property formulation can be achieved [16]. So, for example, by interpreting a trace of length k produced by the simulation model not as a single sample from the *distribution of traces* of length k but instead as a set of k samples from the *distribution of states*, properties about individual states and their likelihood become expressible; by interpreting the trace as a set of $k/2$ samples from the *distribution of subsequent states*, properties about transitions and their likelihood become expressible. In general, a single trace of length k

can be interpreted as a set of samples of trace *fragments* of length $1 \leq i \leq k$. Furthermore, by relating probabilities of individual properties, statements about *correlations of events* can be made. This allows for a high level of granularity and expressivity with respect to property formulation and verification. Technically, the simulator is tightly interwoven with both the modelling framework and the monitor (described below). At the current stage, all simulation replications are executed sequentially. However, since the individual replications are entirely independent, the framework is efficiently parallelisable.

Monitor: The runtime monitor is the central component of the verification framework. Its main purpose is to observe the execution of a single trace as generated by the simulator and check its correctness against the background of a given property on-the-fly, i.e. while the trace is being produced. In the case of thousands of traces that need to be assessed, online verification represents a critical advantage: as soon as a property can be satisfied or violated, the monitor is able to produce a verdict and move on to the next trace. For properties that are satisfiable or refutable at some point along the trace, this leads to significant improvements in speed over an exhaustive approach. As indicated in Sect. 4, the core of a monitor is a temporal formula; it is constructed from the temporal property provided by the user by exploiting expansion laws. The monitor is written in Haskell. Apart from the code being close to the mathematical description of the algorithms, an important decision for choosing Haskell as the underlying programming language was its inherent support for *lazy evaluation*. Given the potentially considerable complexity of the underlying simulation, unnecessary computation is to be avoided in any case. Against this background, it is, for example, important to postpone calls from the monitor to the underlying simulator until a new state is strictly required for evaluating the current property (as defined by the expansion laws). Furthermore, it is important to keep the underlying simulation strictly *forward-oriented*, i.e. such that ticks are simulated in ascending order only and no tick is simulated twice. In that context, Haskell's lazy evaluation strategy is of great help. To illustrate this, consider the problem of evaluating a property ψ on fragments of a trace π. In an offline setting, π would have to be constructed in its entirety prior to evaluation. If ψ is either satisfiable or refutable on a prefix of π, computation would be wasted. In order to avoid that, π must not be produced prior to evaluation. To this end, instead of holding a sequence of global states (which it would if π was the result of a full simulation run), π holds a sequence of *thunks*, i.e. not yet evaluated expressions. Thanks to Haskell's lazy evaluation strategy, a thunk is only evaluated if strictly necessary. In this way, the simulation of the next time step can be postponed in order to achieve the desired online effect. Furthermore, lazy evaluation supports easy *sharing* of computations. Each tick is thus only simulated once which achieves the strict monotonicity effect mentioned above.

5.3 Performance Evaluation

In this section, we provide a brief empirical performance evaluation of MC^{2}MABS. A more detailed evaluation can be found elsewhere [1,13]. We use as our example model a simple disease transmission scenario in which each agent can be either *susceptible*, *infected*, or *recovered*. We assume that transitions between the states are probabilistic and hardcoded and agents are entirely independent. We are aware that, as a consequence of these simplifying assumptions, the model could well be analysed analytically. Choosing an approximate approach may thus seem unnecessarily limiting here. However, it is *not* our goal to perform a realistic verification experiment here. We rather aim to illustrate the effects of *online verification* — the central aspect of our approach — on the performance of the tool. To this end, we have deliberately chosen a simple model. Since `MC`^{2}`MABS` is completely agnostic about the internals of the underlying model and solely focussed on the resulting *traces*, the simplicity of the model does not negatively impact the evaluation results. More comprehensive case studies that focus on the *usefulness* of `MC`^{2}`MABS` for the analysis of swarm-robotic scenarios can be found elsewhere [13,15]. For the evaluation, we focus on the following four properties:

1. '**F** *allInf*': unquantified group property, not refutable before the end of the trace
2. '**G** *allInf*': unquantified group property, immediately refutable
3. '**F**($\forall$ *inf*)': quantified group property, not refutable before the end of the trace
4. '**G**($\forall$ *inf*)': quantified group property, immediately refutable

'*allInf*' describes a population-level proposition that is true if and only if all agents in the population are infected. '*inf*' describes an individual proposition that is true if and only if the agent under consideration is infected. Due to the use of 'finally' and 'globally', respectively, Properties 1 and 2 differ in terms of their *satisfiability*: Property 2 is immediately refutable, whereas the satisfiability of Property 1 can only be determined at the end of the trace. As shown below, this has a significant impact on the time needed for verification. The same is true for Properties 3 and 4.

Properties 1 and 3 as well as Properties 2 and 4 are semantically equivalent; they only differ in terms of their *observational level*: Properties 1 and 3 make a statement about the population as a whole, whereas Properties 2 and 4 are *individual* in nature; the distinction is only made to show the impact of quantification on performance.

We assess the performance against two dimensions: *runtime* and *memory consumption*. Since the executable binary file of `MC`^{2}`MABS` is a merge of code written in both C++ and Haskell, their independent evaluation is not easily possible. For that reason, all individual measurements have to be derived from the profiling results of the entire application. We focus on four major tasks:

Simulation (SIM): Time spent on executing the model logic; this describes the performance of the C++-based simulator.

Extraction (EXT): Time spent on extracting and transforming (*marshalling*) the group traces created by the C++ simulator into their corresponding Haskell vectors.

Verification (VER): Time spent on the actual evaluation of the simLTL property.

Other (OTH): Time spent on 'housekeeping', i.e. other, non simulation- or evaluation-related tasks such as garbage collection, system calls and profiling itself.

Total (TOT): Total runtime of MC²MABS.

The evaluation was performed using `gprof`, Haskell's built-in profiling system on a 64 Bit Dell Latitude Laptop with two Intel® Core^{TM} 2 Duo CPUs (2.8 GHz each), 8 GB of memory and Linux Mint Rebecca (kernel version 3.13.0–24) as operating system. The numbers are averaged over 10 runs, each of which involves the execution of 100 individual simulation runs for the purpose of probability estimation. The results w.r.t. runtime consumption are shown in Table 1, the key points are briefly summarised below.

Table 1. Runtime consumption (in seconds) for different population sizes

	10 agents					100 agents					1,000 agents				
Prop	SIM	VER	EXT	OTH	TOT	SIM	VER	EXT	OTH	TOT	SIM	VER	EXT	OTH	TOT
1	0.022	0.023	0.043	0.012	0.100	0.185	0.081	0.383	0.056	0.706	2.480	0.675	4.361	2.510	10.027
2	0.000	0.006	0.001	0.003	0.010	0.001	0.004	0.003	0.002	0.010	0.021	0.025	0.045	0.029	0.120
3	0.025	0.040	0.042	0.014	0.121	0.242	0.119	0.517	0.079	0.957	2.798	0.715	4.902	2.798	11.213
4	0.001	0.006	0.000	0.002	0.010	0.001	0.004	0.003	0.002	0.010	0.022	0.027	0.036	0.025	0.110

- MC²MABS scales linearly with the size of the underlying population.
- Satisfiability of the formula has a large impact on the time spent on each task. Consider, for example, formula '**G** *allInf*' which is immediately refutable; in this case, evaluation is very quick, even for large populations.
- As the population size grows, an increasing fraction of time is spent on extraction (i.e. marshalling the data structures between C++ and Haskell) and housekeeping, in particular garbage collection. This may represent a bottleneck for large populations which we aim to address as part of the future work.
- MC²MABS also scales linearly with the population size in the case of universally quantified formulae. However, housekeeping (particularly garbage collection) becomes a serious overhead as the number of agents grows. We plan to address this issue in the future, e.g. by employing strictness in some of the operations.

Profiling memory consumption for a lazy language like Haskell can be difficult. For example, expressions without arguments, so-called *Constant Application Forms (CAFs)*, are evaluated only once and shared for later use. Due to their global scope, CAFs are thus, strictly speaking, not part of the call graph and hence need to be treated differently. Straightforward analysis of memory allocated within the call graph only can thus be misleading. In the analysis below, all CAFs are aggregated under the 'Other' section. The results are shown in Table 2. The key points are briefly summarised below.

Table 2. Memory allocation (in Bytes) for different population sizes

	10 agents					100 agents					1,000 agents				
Pr	SIM	VER	EXT	OTH	TOT	SIM	VER	EXT	OTH	TOT	SIM	VER	EXT	OTH	TOT
1	6.4e5	9.7e6	4.5e7	5.7e7	5.7e7	6.4e5	9.7e6	4.4e8	4.5e8	4.5e8	6.4e5	9.7e6	4.4e9	4.4e9	4.4e9
2	6.4e3	1.0e6	4.5e5	3.0e6	3.0e6	6.4e3	1.0e6	4.4e6	6.9e6	6.9e6	6.4e3	1.0e6	4.4e7	4.7e7	4.7e7
3	6.4e5	2.5e7	4.5e7	7.3e7	7.3e7	6.4e5	2.5e7	4.4e8	4.7e8	4.7e8	6.4e5	2.5e7	4.4e9	4.4e9	4.4e9
4	6.4e3	1.2e6	4.5e5	3.4e6	3.4e6	6.4e3	1.2e6	4.4e6	7.3e6	7.3e6	6.4e3	1.2e6	4.4e7	4.7e7	4.7e7

- Memory consumption for both verification and simulation is constant and memory consumption for extraction and marshalling increases linearly with population size.
- Verification is the only evaluation step that the formula size has an impact on, which coincides with the runtime behaviour.

It is important to note that total memory allocation is not sufficient for understanding the full allocation behaviour of the program. It is useful to also analyse the *runtime heap profile* which describes memory allocation *over time*. An example for 1,000 agents (Property 3) is shown in Fig. 2. For clarity, we restrict the number of functions and data structures shown to 10. In the graph, 'Pinned objects' refers to information in memory which is not movable by the garbage collection, e.g. memory allocated in the C++ part of the application. Furthermore, some of the functions in the Haskell implementation are split up into an inner and an outer part for technical reasons; this distinction is also reflected in the graphs. Finally, due to the pure functional nature of Haskell, global state cannot be maintained. In order to emulate this functionality, alternative options such as the *State Monad* have to be used. This state management accounts for a certain level of memory allocation which is also considered in the analysis. The graph shows that the peak memory allocation is stable and fairly low compared with the overall memory consumption; the graphs also show that the amount of garbage collection (indicated by the reduction in memory consumption) is clearly a function of the runtime.

6 Case Studies

For space constraints, we cannot provide a full case study here. An exhaustive evaluation of three different versions of the swarm foraging scenario introduced in Sect. 2 including source code is provided in the first author's PhD thesis [13]. Another evaluation of the same scenario with a particular focus on the quantitative capabilities of MC^{2}MABS is provided in [16]. The paper illustrates how properties about transition probabilities, residence probabilities, state distributions, and correlations between events can be formulated and answered purely based on the analysis of individual traces.

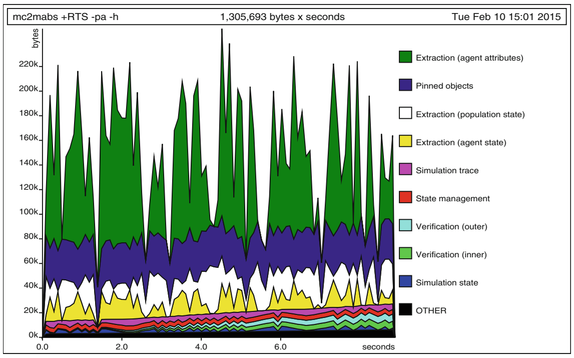


Fig. 2. Runtime heap profile for 1,000 agents

7 Conclusions and Future Work

This paper described the architecture, design decisions and implementation details of `MC`^{2}`MABS`, a statistical runtime verification framework for ABSs. With properties formulated in temporal logic, high scalability due to the focus on individual and independent simulation traces and its ability to provide confidence intervals for the results, it aims to combine some of the strengths of both formal and informal verification techniques into a common framework. The approach aims to satisfy the four requirements introduced in Sect. 2 as follows:

Efficiency: Due to the approximate nature of the approach, simulation models with a large number of constituents are efficiently verifiable. The number of simulation runs necessary for verification is only dependent on the desired level of precision and not on the size of the underlying system. Since individual traces are entirely independent, the approach is also inherently parallel and therefore highly scalable.

Expressivity: Properties can be formulated in temporal logic and checked automatically. The syntax of the specification language provides quantification and selection operators and thus allows for the formulation of properties on different observational levels. In addition to that, the trace fragment-based semantics allow for the verification of quantitative properties.

Flexibility: Due to the reliance on simulation traces rather than on a formal model, arbitrary simulation models are verifiable.

Immediacy: Verification is performed upon the output of the original simulation.

It is important to stress here that we do not aim to propose a substitute for purely formal verification. In cases where formal verification is feasible, it is clearly preferable over an approximate approach such as the one describe here. However, for cases which are not (and may never be) formally verifiable, `MC`^{2}`MABS` can present an interesting alternative and help to gain a better understanding of the emergent simulation dynamics than is currently possible. First experiments produced encouraging results and showed that the tool allows for the verification of complex simulation models with a high level of confidence ($> 99\,\%$) and accuracy ($< 1\,\%$) in a timely manner [13,15]. Before `MC`^{2}`MABS` can be used efficiently in the real world, there are, of course, still plenty of limitations and open problems to be overcome. Some are mentioned below.

Accuracy: It is clear that, for highly safety-critical areas, a significantly higher level of precision than that used in our experiments is needed. With the current estimation procedure, the number of simulation traces increases quadratically with the level of accuracy which represents a critical bottleneck. One way to remedy this problem is to use ideas from *rare event sampling* in order to reduce the sample size needed.

Efficiency: An important advantage of trace-based verification is that it is efficiently parallelisable. At the current stage, `MC`^{2}`MABS` performs verification sequentially and does not exploit the capabilities of modern parallel hardware. A second important starting point for performance improvements is the performance of the simulation itself. For example, by making use of C++ template metaprogramming, some of the runtime calculations can be shifted towards compile time. Finally, `MC`^{2}`MABS` is technically subdivided into a simulation (written in C++) and a verification part (written in Haskell). Marshalling, i.e. translating and transferring the data structures between the two languages represents a significant bottleneck which also negatively influences the capability of the tool to analyse large batches of simulation traces.

Usability: The tool is still in a prototypical state and its usability is therefore still fairly low. Temporal logic and C++ are certainly not the most typical skills of an agent-based modeller. The choice has been made for the purpose of rigour and performance. But it is clear that, if the tool is to be used practically, higher-level interfaces need to be developed. The same holds for the connection to existing simulators such as Repast or NetLogo which currently requires manual efforts.

References

1. MC^{2}MABS. https://github.com/bherd/mc2mabs. Accessed February 2015
2. Baier, C., Katoen, J.-P.: Principles of Model Checking. The MIT Press, Cambridge (2008)
3. Ballarini, P., Fisher, M., Wooldridge, M.: Uncertain agent verification through probabilistic model-checking. In: Barley, M., Mouratidis, H., Unruh, A., Spears, D., Scerri, P., Massacci, F. (eds.) Safety and Security in Multiagent Systems. LNCS, vol. 4324, pp. 162–174. Springer, Heidelberg (2009)
4. Bharathy, G.K., Silverman, B.: Validating agent based social systems models. In: Proceedings of the Winter Simulation Conference, pp. 441–453. Winter Simulation Conference (2010)
5. Bosse, T., Mogles, N.: Comparing modelling approaches in aviation safety. In: Curran, R. (eds.) Proceedings of the 4th International Air Transport and Operations Symposium, Toulouse, France (2013)

6. Çakırlar, İ., Gürcan, Ö., Dikenelli, O., Bora, Ş.: RatKit: repeatable automated testing toolkit for agent-based modeling and simulation. In: Grimaldo, F., Norling, E. (eds.) MABS 2014. LNCS, vol. 9002, pp. 17–27. Springer, Heidelberg (2015)
7. Cao, Y.U., Fukunaga, A.S., Kahng, A.: Cooperative mobile robotics: antecedents and directions. Auton. Robots **4**(1), 7–27 (1997)
8. Collier, N.: Repast: an extensible framework for agent simulation. Nat. Resour. Environ. Issues **8**, 17–21 (2001)
9. Dastani, M., Hindriks, K.V., Meyer, J.-J.: Specification and Verification of Multi-Agent Systems. Springer Science & Business Media, Heidelberg (2010)
10. De Wolf, T., Holvoet, T., Samaey, G.: Development of self-organising emergent applications with simulation-based numerical analysis. In: Brueckner, S.A., Di Marzo Serugendo, G., Hales, D., Zambonelli, F. (eds.) ESOA 2005. LNCS (LNAI), vol. 3910, pp. 138–152. Springer, Heidelberg (2006)
11. Dekhtyar, M.I., Dikovsky, A.J., Valiev, M.K.: Temporal verification of probabilistic multi-agent systems. In: Avron, A., Dershowitz, N., Rabinovich, A. (eds.) Pillars of Computer Science. LNCS, vol. 4800, pp. 256–265. Springer, Heidelberg (2008)
12. Hérault, T., Lassaigne, R., Magniette, F., Peyronnet, S.: Approximate probabilistic model checking. In: Steffen, B., Levi, G. (eds.) VMCAI 2004. LNCS, vol. 2937, pp. 73–84. Springer, Heidelberg (2004)
13. Herd, B.: Statistical runtime verification of agent-based simulations. PhD thesis, King's College London (2015)
14. Herd, B., Miles, S., McBurney, P., Luck, M.: An LTL-based property specification language for agent-based simulation traces. TR 14–02, King's College London, October 2014
15. Herd, B., Miles, S., McBurney, P., Luck, M.: Approximate verification of swarm-based systems: a vision and preliminary results. In: Engineering Systems for Safety: Proceedings of 23rd Safety-Critical Systems Symposium. CreateSpace Independent Publishing Platform (2015)
16. Herd, B., Miles, S., McBurney, P., Luck, M.: Towards quantitative analysis of multiagent systems through statistical model checking. In: 3rd International Workshop on Engineering Multiagent Systems (2015)
17. Karnouskos, S., de Holanda, T.: Simulation of a smart grid city with software agents. In: 3rd European Symposium on Computer Modeling and Simulation, pp. 424–429, November 2009
18. Konur, S., Dixon, C., Fisher, M.: Formal verification of probabilistic swarm behaviours. In: Dorigo, M., Birattari, M., Di Caro, G.A., Doursat, R., Engelbrecht, A.P., Floreano, D., Gambardella, L.M., Groß, R., Şahin, E., Sayama, H., Stützle, T. (eds.) ANTS 2010. LNCS, vol. 6234, pp. 440–447. Springer, Heidelberg (2010)
19. Konur, S., Dixon, C., Fisher, M.: Analysing robot swarm behaviour via probabilistic model checking. Robot. Auton. Syst. **60**(2), 199–213 (2012)
20. Kwiatkowska, M., Lomuscio, A., Qu, H.: Parallel model checking for temporal epistemic logic. In: Proceedings of 19th European Conference on Artificial Intelligence, pp. 543–548, IOS Press (2010)
21. Legay, A., Delahaye, B., Bensalem, S.: Statistical model checking: an overview. In: Barringer, H., Falcone, Y., Finkbeiner, B., Havelund, K., Lee, I., Pace, G., Roşu, G., Sokolsky, O., Tillmann, N. (eds.) RV 2010. LNCS, vol. 6418, pp. 122–135. Springer, Heidelberg (2010)
22. Leucker, M., Schallhart, C.: A brief account of runtime verification. J. Logic Algebraic Program. **78**(5), 293–303 (2009)
23. Liu, W., Winfield, A., Sa, J.: Modelling swarm robotic systems: a case study in collective foraging. In: Towards Autonomous Robotic Systems, pp. 25–32 (2007)

24. Liu, W., Winfield, A.F.T., Sa, J., Chen, J., Dou, L.: Strategies for energy optimisation in a swarm of foraging robots. In: Şahin, E., Spears, W.M., Winfield, A.F.T. (eds.) SAB 2006 Ws 2007. LNCS, vol. 4433, pp. 14–26. Springer, Heidelberg (2007)
25. Lomuscio, A., Penczek, W., Qu, H.: Partial order reductions for model checking temporal-epistemic logics over interleaved multi-agent systems. Fundamenta Informaticae **101**(1–2), 71–90 (2010)
26. Lomuscio, A., Penczek, W., Wożna, B.: Bounded model checking for knowledge and real time. Artif. Intell. **171**(16–17), 1011–1038 (2007)
27. Lorscheid, I., Heine, B.-O., Meyer, M.: Opening the 'black box' of simulations: increased transparency and effective communication through the systematic design of experiments. Comput. Math. Organ. Theor. **18**(1), 22–62 (2012)
28. Macal, C.M., North, M.J.: Tutorial on agent-based modelling and simulation. J. Simul. **4**(3), 151–162 (2010)
29. Marks, R.E.: Validating simulation models: a general framework and four applied examples. Comput. Econ. **30**, 265–290 (2007)
30. McCune, R., Madey, G.: Agent-based simulation of cooperative hunting with UAVs. In: Proceedings of the Agent-Directed Simulation Symposium. Society for Computer SimulationInternational (2013)
31. McKelvey, B.: The blackwell companion to organizations, chapter model-centered organization science epistemology, pp. 752–780. Blackwell (2002)
32. Niazi, M., Hussain, A., Kolberg, M.: Verification and validation of agent based simulations using the VOMAS approach. In: Proceedings of 3rd Workshop on Multi-Agent Systems and Simulation (2009)
33. Pedersen, T., Dyrkolbotn, S.K.: Agents homogeneous: a procedurally anonymous semantics characterizing the homogeneous fragment of ATL. In: Boella, G., Elkind, E., Savarimuthu, B.T.R., Dignum, F., Purvis, M.K. (eds.) PRIMA 2013. LNCS, vol. 8291, pp. 245–259. Springer, Heidelberg (2013)
34. Phan, D., Varenne, F.: Agent-based models and simulations in economics and social sciences: from conceptual exploration to distinct ways of experimenting. J. Artif. Soc.Soc. Simul. **13**(1), 5 (2010)
35. Sargent, R.G.: Verification and validation of simulation models. In: Proceedings of 40th Winter Simulation Conference (WSC 2008), pp. 157–169 (2008)
36. Sebastio, S., Vandin, A.: MultiVeStA: statistical model checking for discrete event simulators. In: Proceedings of 7th International Conference on Performance Evaluation Methodologies and Tools (2013)
37. Wan, W., Bentahar, J., Ben Hamza, A.: Model checking epistemic and probabilistic properties of multi-agent systems. In: Mehrotra, K.G., Mohan, C.K., Oh, J.C., Varshney, P.K., Ali, M. (eds.) IEA/AIE 2011, Part II. LNCS, vol. 6704, pp. 68–78. Springer, Heidelberg (2011)
38. Wilensky, U.: NetLogo. TR, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1999)
39. Wright, C.J., McMinn, P., Gallardo, J.: Towards the automatic identification of faulty multi-agent based simulation runs using MASTER. In: Giardini, F., Amblard, F. (eds.) MABS 2012. LNCS, vol. 7838, pp. 143–156. Springer, Heidelberg (2013)

Data Driven Validation Framework for Multi-agent Activity-Based Models

Jan Drchal(✉), Michal Čertický, and Michal Jakob

Faculty of Electrical Engineering, Czech Technical University in Prague, Prague, Czech Republic
drchajan@fel.cvut.cz, {certicky,jakob}@agents.fel.cvut.cz

Abstract. Activity-based models, as a specific instance of agent-based models, deal with agents that structure their activity in terms of (daily) activity schedules. An activity schedule consists of a sequence of activity instances, each with its assigned start time, duration and location, together with transport modes used for travel between subsequent activity locations. A critical step in the development of simulation models is validation. Despite the growing importance of activity-based models in modelling transport and mobility, there has been so far no work focusing specifically on statistical validation of such models. In this paper, we propose a six-step *Validation Framework for Activity-based Models (VALFRAM)* that allows exploiting historical real-world data to assess the validity of activity-based models. The framework compares temporal and spatial properties and the structure of activity schedules against real-world travel diaries and origin-destination matrices. We confirm the usefulness of the framework on three activity-based transport models.

1 Introduction

Transport and mobility have recently become a prominent application area for multi-agent systems and agent-based modelling (Chen and Cheng 2010). Models of transport systems offer an objective common ground for discussing policies and compromises (de Dios Ortúzar and Willumsen 2011), help to understand the underlying behaviour of these systems and aid in the actual decision making and transport planning.

Large-scale, complex transport systems, set in various socio-demographic contexts and land-use configurations, are often modelled by simulating the behaviour and interactions of millions of autonomous, self-interested agents. Agent-based modelling paradigm generally provides a high level of detail and allows representing non-linear patterns and phenomena beyond traditional analytical approaches (Bonabeau 2002). Specific subclass of agent-based models, called *activity-based models*, address particularly the need for realistic representation of travel demand and transport-related behaviour. Unlike traditional trip-based models, activity-based models view travel demand as a consequence of agent's needs to pursue various activities distributed in space and understanding of travel decisions is secondary to a fundamental understanding of activity behaviour (Jones et al. 1990).

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 55–67, 2016.
DOI: 10.1007/978-3-319-31447-1_4

Gradual methodological shift towards such a behaviourally-oriented modelling paradigm is evident. An early work on the topic is represented by the CARLA model, developed as a part of the first comprehensive assessment of behaviourally-oriented approach at Oxford (Jones et al. 1983). Later work is represented by the SCHEDULER model – a cognitive architecture producing activity schedules from long- and short-term calendars and perceptual rules (Gärling et al. 1994) or ALBATROSS, which was the first model where the activity scheduling process was completely automatically estimated from the data (Arentze and Timmermans 2000).

In order to produce dependable and useful results, the model needs to be *valid*[1] enough. In fact, validity is often considered the most important property of models (Klügl 2009). The process of quantifying the model validity by determining whether the model is an accurate representation of the studied system is called *validation* and the validation process needs to be done thoroughly and throughout all phases of model development (Law 2009).

Despite the growing adoption of activity-based models and the generally acknowledged importance of model validation, a validation process for activity-based models in particular has not yet been standardized by a detailed methodological framework. Validation techniques and guidelines are addressed in most modelling textbooks (Law 2007) and have even been instantiated in the form of a validation process for general agent-based models (Klügl 2009); however, such techniques are still too general to provide concrete, practical methodology for the key validation step: statistical validation against real-world data.

In this paper, we address this gap and propose a validation framework entitled VALFRAM (Validation Framework for Activity-based Models), designed specifically for statistically quantifying the validity of *activity-based* transport models. The framework relies on the real-world transport behaviour data and quantifies the model validity in terms of clearly defined validation metrics. We illustrate and demonstrate the framework on several activity-based transport models of a real-world region populated by approximately 1 million citizens.

2 Preliminaries

2.1 Activity-Based Models

Activity-based models (Ben-Akivai et al. 1996) are multiagent models in which the agents plan and execute so-called *activity schedules* – finite sequences of *activity instances* interconnected by *trips*. Each activity instance needs to have a specific *type* (e.g., `work`, `school` or `shop`), *location*, desired *start time* and *duration*. Trips between activity instances are specified by their main transport mode (e.g., `car` or `public transport`).

[1] *Valid* model is a model of sufficient *accuracy*. We use these terms interchangeably in the following text.

2.2 Validation Methods

Validation methods in general are usually divided into two types:

- *Face validation* subsumes all methods that rely on natural human intelligence such as expert assessments of model visualizations. Face validation shows that model's behaviour and outcomes are reasonable and plausible within the frame of the theoretic basis and implicit knowledge of system experts or stake-holders. Face validation is in general incapable of producing quantitative, comparable numeric results. Its basis in implicit expert knowledge and human intelligence also makes it difficult to standardize face validation in a formal methodological framework. In this paper, we therefore focus on statistical validation.
- *Statistical validation* (sometimes called *empirical*) employs statistical measures and tests to compare key properties of the model with the data gathered from the modelled system (usually the original real-world system).

From a higher-level perspective, VALFRAM can be viewed as an activity-based model-focused implementation of the *statistical validation* step of a more comprehensive validation procedure for generic agent-based models, introduced in (Klügl 2009), as depicted in Fig. 1. Besides the face and statistical validation, this procedure features other complementary steps such as calibration and sensitivity analysis.

Being set in the context of activity-based modelling, the VALFRAM framework is concerned with the specific properties of activity schedules generated by the agents within the model. These properties are compared to historical real-world data in order to compute a set of numeric similarity metrics.

3 VALFRAM Description

In this section a detailed description of VALFRAM is given. We cover validation data, validation objectives and finally measures defined by VALFRAM.

3.1 Data

A requirement for statistical validation of any model is data capturing the relevant aspects of the behaviour of modelled system, against which the model is validated. To validate an activity-based model, the VALFRAM framework requires two distinct data sets gathered in the modelled system:

1. *Travel Diaries*: Travel diaries are usually obtained by long-term surveys (taking up to several days), during which participants log all their trips. The resulting data sets contain anonymized information about every participant (usually demographic attributes such as age, gender, etc.), and a collection of all their trips with the following properties: *time* and *date, duration, transport mode(s)* and *purpose* (the activity type at the destination). More detailed travel diaries also contain the *locations* of the origin and the destination of each trip.

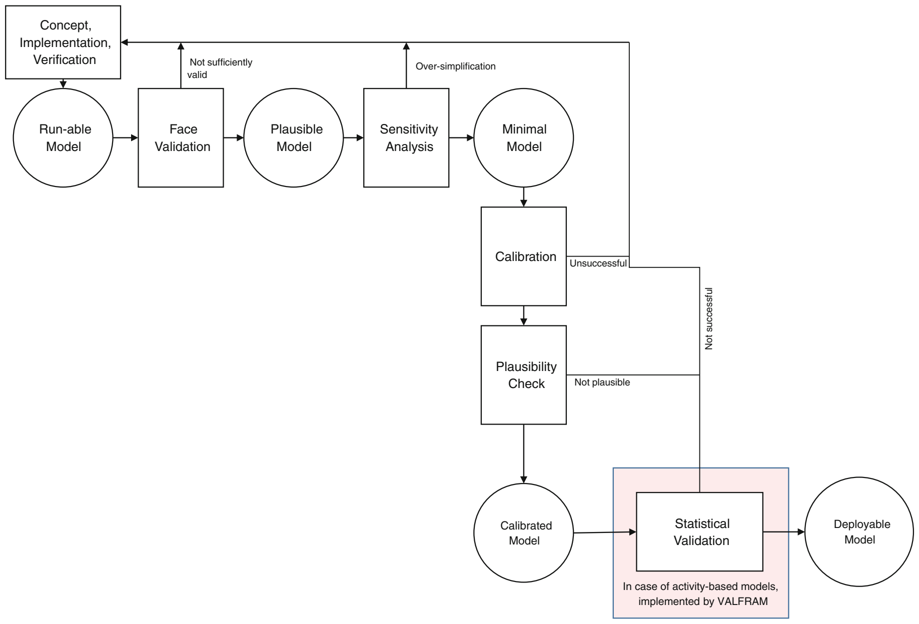


Fig. 1. Higher-level validation procedure for agent-based models in general, introduced in (Klügl 2009). VALFRAM implements the statistical validation step specifically for activity-based models.

2. *Origin-Destination Matrix* (O-D Matrix): The most basic O-D matrices are simple 2D square matrices displaying the number of trips travelled between every combination of origin and destination locations during a specified time period (e.g., one day or one hour). The origin and destination locations are usually predefined, mutually exclusive zones covering the area of interest and their size determines the level of detail of the matrix.

3.2 VALFRAM Validation Objectives

The VALFRAM validation framework is concerned with a couple of specific properties of activity schedules produced by modelled agents. These particular properties need to correlate with the modelled system in order for the model to accurately reproduce the system's transport-related behaviour. At the same time, these properties can actually be validated based on available data sets – travel diaries and O-D matrices. In particular, we are interested in:

A. *Activities* and their:
   1. *temporal* properties (start times and durations),
   2. *spatial* properties (distribution of activity locations in space),
   3. *structure* of activity sequences (typical arrangement of successive activity types).

B. *Trips* and their:
   1. *temporal* properties (transport mode choice in different times of day; durations of trips),
   2. *spatial* properties (distribution of trip's origin-destination pairs in space),
   3. *structure* of transport mode choice (typical mode for each destination activity type).

3.3 VALFRAM Validation Metrics

To validate these properties of interest, we need to perform six validation steps (A1, A2, A3, B1, B2, B3), as depicted in Table 1 and detailed in the rest of this section. In each validation step, we compute specific numeric metrics (statistics). For all metrics, higher values of these statistics indicate a larger difference between the model and validation set, i.e., lower accuracy.

Table 1. Six validation steps of VALFRAM framework and corresponding validation data sets needed for each of them.

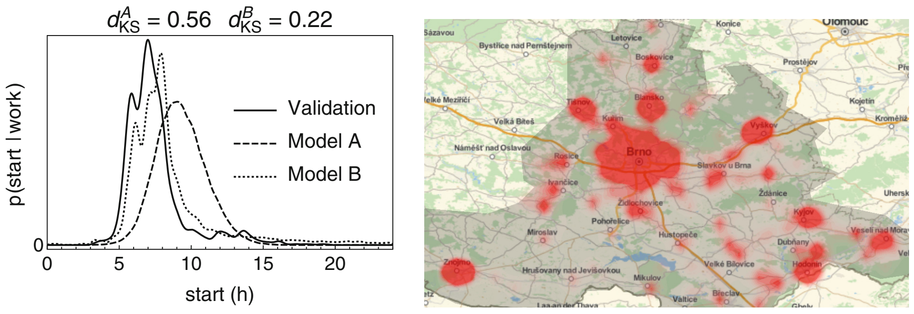
	A. Activities		B. Trips	
	Task	Data set	Task	Data set
1. Time	Compare the distributions of *start times* and *durations* for each activity type using Kolmogorov-Smirnov (KS) statistic.	Travel Diaries	Compare the distribution of selected *modes* by *time of day* and the distribution of *travel times* by *mode* using χ^2 and KS statistics.	Travel Diaries
2. Space	Compare distribution of each activity type in 2D space using RMSE. Plot heat maps for additional feedback.	Space-aware Travel Diaries	Compute the distance between generated and real-world O-D matrix using RMSE.	Origin-Destination Matrix
3. Structure	*(i)* Compare activity counts within activity schedules using χ^2 statistics. *(ii)* Compare distributions of activity schedule subsequences as n-grams profiles using χ^2 statistics.	Travel Diaries	Compare the distribution of selected *transport mode* for each type of *target activity* type using χ^2 statistics.	Travel Diaries

A1. Activities in Time: The comparison of activity distributions in time is realized by means of a well-established Kolmogorov-Smirnov two-sample statistic[2] (Hollander et al. 2013). VALFRAM applies the method to start time distributions $p(\text{start}|\text{act. type})$ as well as to duration distributions $p(\text{duration}|\text{act. type})$.

[2] We have also experimented with related Anderson-Darling and Cramér–von Mises statistics getting similar results. Kolmogorv-Smrnov was finally selected as it is widely known and easier to get insight into.

The statistic is defined as the maximum deviation between the empirical cumulative distribution functions F_M and F_V which are based on the model and validation data distributions: $d_{KS} = \sup_x |F_M(x) - F_V(x)|$. The values lie in the interval $[0, 1]$.

Figure 2a shows an example application of the Kolmogorov-Smirnov statistic comparing two different models to validation data.



(a) `work` activity start time (b) Modeled area, `sleep` activity

Fig. 2. Start time distributions for `work` activity shown for validation data and two different models (a) including Kolmogorov-Smirnov statistics. Modelled area including `sleep` activity spatial PDF visualized as a heat map (b).

A2. Activities in Space: The comparison of activity distributions in space is performed separately for every activity type. Unlike in the previous step, the distributions are two-dimensional (latitude, longitude or projected coordinates). The process consists of the following steps. First, bivariate empirical cumulative distribution functions (ECDFs) F_M and F_V are constructed using coordinate data for both model and validation data, respectively. Second, F_M and F_V are regularly sampled getting matrices E^M and E^V both having m rows and n columns. Third, Root Mean Squared Error (RMSE) of the two matrices is computed using

$$d_{ecdf} = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} \left(E^M_{ij} - E^V_{ij}\right)^2 / (mn)}. \tag{1}$$

As $E^M_{ij} \leq 1$ and $E^V_{ij} \leq 1$, the measure d_{ecdf} is again limited to the $[0, 1]$ interval.

Figure 2b shows the spatial probability distribution function (PDF) of `sleep` type activities on the validation set visualized as a heat map. The probability distribution was approximated from data using Gaussian kernels. Similar heat maps might be helpful when developing a model as they can show where problems or imprecisions are.

A3. Structure of Activities: In the previous steps, we examined the activity distributions in time and space. In this step, we consider the activity composition of the entire activity schedules. We propose a measure which compares distributions of *activity counts* in activity schedules as well as a measure comparing the distribution of possible *activity type sequences*.

Activity Count: The comparison of activity counts in activity schedules is based on a well-known Pearson's chi-square test (Sokal and Rohlf 1994). The procedure is performed separately for each activity type. First, frequencies f_i^M and f_i^V for the count i are collected for both model and validation data. Validation data frequencies f_i^V are then used to get count proportions p_i^V and in turn validation frequencies s_i^V scaled to match the sum of model's frequencies ($\sum_i s_i^V = \sum_i f_i^M$). Using f_i^M and s_i^V chi-square statistic is computed as

$$\chi^2 = \sum_i \left(f_i^M - s_i^V\right)^2 / s_i^V . \tag{2}$$

Activity Sequences: We also compare activity sequence distributions. The method is based on the well-established text mining techniques (Manning 1999). Particularly, we compare *n-gram profiles* using chi-square statistic. N-gram is a continuous subsequence of the original sequence having a length exactly n. Consider an example activity schedule consisting of the following activity sequence: $\langle \texttt{none}, \texttt{sleep}, \texttt{work}, \texttt{leisure}, \texttt{sleep}, \texttt{none} \rangle$[3]. The set of all 2-grams (bigrams) is then: $\{\langle \texttt{none}, \texttt{sleep}\rangle, \langle \texttt{sleep}, \texttt{work}\rangle, \langle \texttt{work}, \texttt{leisure}\rangle, \langle \texttt{leisure}, \texttt{sleep}\rangle$ and $\langle \texttt{sleep}, \texttt{none}\rangle\}$. We create an *n-gram profile* by counting frequencies of all n-grams in a range $n \in \{1, 2, \cdots, k\}$ for all activity schedules. All the N n-grams are then sorted by their counts in a decreasing order so that the counts are $f_i \geq f_j$ for any two n-grams i and j where $1 \leq i < j \leq N$ (for a tie $f_i = f_j$ one should sort in the lexicographical order). We only work with a proportion P of n-grams having the highest count in the profile. More precisely, we take only the first M n-grams, where M is the highest value for which $\sum_{i=1}^{M} f_i \leq P \sum_{i=1}^{N} f_i$ is true.

In order to compare n-gram profiles of model and validation data, we employ chi-square statistic matching both profiles by the corresponding n-grams (only n-grams found in both profiles are considered).

B1. Trips in Time: The validation of trips in time consists of two sub-steps: a comparison of mode distributions for a given time of day and a comparison of travel time distributions for selected modes.

Modes by Time of Day: The comparison of mode distributions for a given time of day, i.e., $p(\text{mode}|\text{time range})$, is based on exactly the same approach which we used to compare activity counts (validation step A3): the χ^2 statistic is computed for mode frequencies of trips starting in a selected time interval. We suggest computing χ^2 statistic for twenty four one-hour intervals per day, although other partitionings are possible.

Travel Times per Mode: Travel time distributions for modes $p(\text{travel time}|\text{mode})$ are validated in the same way as activities in time (see validation step A1) using Kolmogorov-Smirnov statistic d_{KS}.

[3] Note, that `none` activities are added to the beginning and end of the activity schedule in order to preserve information about initial/terminal activity.

B2. Trips in Space: In order to validate trip distributions in space, we propose a symmetrical dissimilarity measure based on O-D matrix comparison. The algorithm is realized in three consecutive steps. First, O-D matrices are rearranged to use a common set of origins and destinations. Second, both matrices are scaled to make trip counts comparable. Third, RMSE for all elements which have non zero trip count in either of the matrices is computed.

The algorithm starts with two O-D matrices: model matrix M and validation matrix V. Each element M_{ij} (or V_{ij}) represents a count of trips between origin i and destination j. The positional information (i.e., latitude/longitude or other type of coordinates) is denoted $m_i, m_j \in C_M$ for model and similarly $v_i, v_j \in C_V$ for validation data where C_M and C_V are sets of all possible coordinates (e.g., all traffic network nodes).

Note that in most practical cases $C_M \neq C_V$. As an example we can have precise GPS coordinates generated by the model, however, only approximate or aggregated trip locations from validation travel diaries. As we have to work with the same locations in order to compare the O-D matrices, we need to select a common set of coordinates C. In practice, this would be typically the validation data location set ($C = C_V$) while all locations from C_M must be projected to it by replacing each m_i by its closest counterpart in C. This might eventually lead to resizing of the O-D matrix M as more origins/destinations might get aggregated into a single row/column.

In many cases the total number of trips in M and V can be vastly different. The second step of the algorithm scales both M and V to a total element sum of one: $M'_{ij} = \frac{M_{ij}}{\sum_i \sum_j M_{ij}}$ and $V'_{ij} = \frac{V_{ij}}{\sum_i \sum_j V_{ij}}$. Each element of both M'_{ij} and V'_{ij} now represents a relative traffic volume between origin i and destination j.

Finally, we compute the O-D matrix distance using the following equation:

$$d_{OD} = \sqrt{\frac{\sum_i \sum_j \left(M'_{ij} - V'_{ij}\right)^2}{\left|\left\{(i,j) : M'_{ij} > 0 \vee V'_{ij} > 0\right\}\right|}}. \tag{3}$$

Note that the equation is RMSE computed over all origin-destination pairs which appear either in M'_{ij}, V'_{ij} or in both. We have decided to ignore the elements which are zero in both matrices as these might represent trips which might not be possible at all (i.e., not connected by the transport network). Possible values of d_{OD} lie in interval $[0, 1]$ (the upper bound is given by $M'_{ij} \leq 1$ and $V'_{ij} \leq 1$).

B3. Mode for Target Activity Type: The validation of the mode choice for target activity type is again based on χ^2 statistic. Here, we collect counts per each mode for each target activity of choice.

4 VALFRAM Evaluation

In general, we expect a statistical validation framework to meet three key conditions. First, the framework quantifies the accuracy of the validated models in a way which allows comparing model's accuracy in replicating different aspects of the behaviour of the modelled system. Second, data required for validation are available. Third, validation results produced by the framework correlate with the expectations based on expert insight and face validation.

VALFRAM meets conditions 1 and 2 for activity-based models by explicitly expressing the spatial, temporal and structural properties of activities and trips, using only travel diaries and O-D matrices. To evaluate it with respect to condition 3, we have built three different activity-based models, formulated hypotheses about them based on our expert insight and used VALFRAM to validate both of them.

4.1 Evaluation Models

The first model, denoted $\mathrm{M_A}$ (model A), is a rule-based model inspired by ALBATROSS[4] (Arentze and Timmermans 2000). The second model, denoted $\mathrm{M_B}$, is a fully data-driven model based on Recurrent Neural Networks (RNNs). More specifically, the model employs fully-connected Long-Short Term Memory (LSTM) units (Hochreiter and Schmidhuber 1997) and several sets of softmax output units. Given the training dataset based on travel diaries, the model is trained to repetitively take current activity type and its end time as an input in order to produce a trip (including trip duration and main mode) and the following activity (defined by type and duration). As $\mathrm{M_B}$ is currently unable to generate spatial component of the schedules (e.g., activity locations), VALFRAM steps A2 and B2 are evaluated on a predecessor of $\mathrm{M_A}$ denoted $\mathrm{M'_A}$ (model A′). $\mathrm{M'_A}$ uses a less sophisticated approach to select activity locations.

All $\mathrm{M_A}$, $\mathrm{M_B}$ and $\mathrm{M'_A}$ models were used to generate a sample of 100,000 activity schedules. Our validation set V contained approximately 1,800 schedules. Such a disproportion is typical in reality, since obtaining real-world data tends to be more costly than obtaining synthetic data from model. All the data used in this study cover a single workday. An overview of the modelled area is depicted in Fig. 2b.

In the following text we present five hypotheses based on our insight of models. Note that all VALFRAM steps A1 through B3 are performed in order to evaluate them.

4.2 Test Hypotheses

Hypothesis 1: The rule-based model $\mathrm{M_A}$ uses a very simple linear classifier for decisions on activity start times, so it will likely perform worse than the RNN-based model in their assignment. On the other hand, the activity scheduler in $\mathrm{M_A}$ performs schedule optimization, during which it adapts activity durations according to rules psychologically plausible. This should produce more realistic behaviour than the purely data-driven RNN model[5].

[4] Although we call $\mathrm{M_A}$ the rule-based model, it estimates activity count, durations and occasionally start times using linear-regression models based on data. All other activity schedule properties are based on rules constructed using expert knowledge.

[5] At least given the limited size of the RNN training dataset.

Step A1 of VALFRAM confirms the hypothesis. Figure 3a depicts the distributions $p(\text{start}|\texttt{work})$ for validation data V and models $\mathrm{M_A}$ and $\mathrm{M_B}$. The values $d^A_{KS} > d^B_{KS}$ indicate the higher accuracy of the RNN model, with the most significant difference in the case of `work` and `school` activities. On the other hand, Fig. 3b shows that $\mathrm{M_A}$ outperforms $\mathrm{M_B}$ in terms of activity durations.

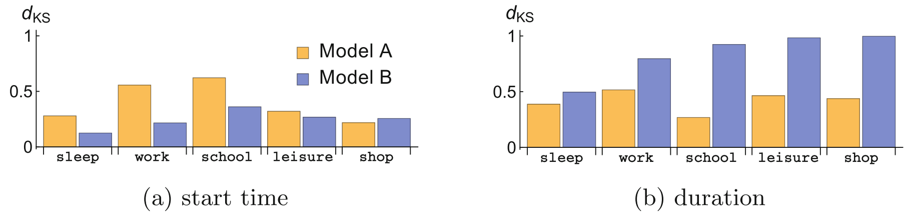


Fig. 3. An example of activity in time comparison. The values of d_{KS} are shown for both models $\mathrm{M_A}$ and $\mathrm{M_B}$. $\mathrm{M_B}$ outperforms $\mathrm{M_A}$ on start times while the situation is the opposite for durations.

Hypothesis 2: Activity sequences of real-world system tend to be harder to replicate using simple rule-based models than robust data-driven approaches.

Results of the step A3 (*activity counts*) for all the activity types are shown in Table 2. The data-driven model $\mathrm{M_B}$ outperforms $\mathrm{M_A}$ with the exception of the `leisure` activity (which we later found to be insufficiently covered by the RNN training data). Note that both $\mathrm{M_A}$ and $\mathrm{M_B}$ give the same χ^2 value for the `sleep` activity which is caused by the fact that both models generate daily schedules having strictly two `sleep` activities in the current setup. For the step A3 (*activity sequences*) we got the following results for both models using the proportion $P = 0.9$ and $k = 11$ (longest sequence in data): $\chi^2 \approx 8.4 \times 10^5$ for $\mathrm{M_A}$ and $\chi^2 \approx 2.6 \times 10^5$ for $\mathrm{M_B}$ showing superiority of the RNN model.

Table 2. Activity counts for selected activities (χ^2 statistic). Model $\mathrm{M_B}$ outperforms model $\mathrm{M_A}$ with the exception of the `leisure` activity type.

Model	`sleep`	`work`	`school`	`leisure`	`shop`
$\mathrm{M_A}$	21468.1	2889.3	542.2	**1750.3**	974.2
$\mathrm{M_B}$	21468.1	**255.7**	**293.8**	4625.7	**773.8**

Hypothesis 3: While rule-based model optimizes the whole daily activity plans, RNN-based model works sequentially and schedules new activity based only on the previous ones. Therefore, it will be less accurate towards the end of the day.

By a further analysis of step A3 (*activity sequences*), which involved the comparison of a set of n-grams having highest frequency difference, we have, indeed, found that the RNN model tends to be less accurate towards the end of the generated activity sequence resulting in schedules not ended by the `sleep` activity in a number of cases. Moreover, Fig. 4 shows a comparison of *mode by time of day* selection χ^2 values (step B1) for M_A and M_B showing that although M_B is initially more accurate it eventually degrades and the rule-based model M_A prevails.

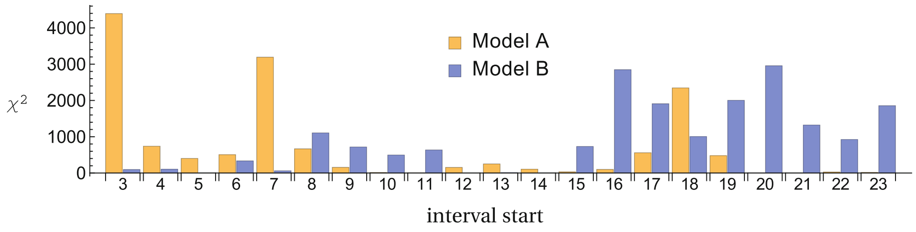


Fig. 4. Modes by the time of day. The figure shows a comparison of χ^2 values for `car` and `public transport` modes for one hour intervals between 3:00 and 23:00.

Hypothesis 4: Unlike the rule-based model, the RNN model has no access to trip-planning data (i.e., transport network, timetables) which will decrease its performance in selecting trip modes.

For the step B1 (*travel times per mode*) we got $d^A_{KS} = 0.22 < d^B_{KS} = 0.31$ for `car` and $d^A_{KS} = 0.37 < d^B_{KS} = 0.43$ for `public transport` modes. Results of the step B3 are summarized in Table 3 also supporting the superiority of M_A in modelling mode selection.

Table 3. Transport mode selection for target activity type (χ^2 statistic). Model M_A outperforms model M_B in four out of five activity types.

Model	`sleep`	`work`	`school`	`leisure`	`shop`
M_A	**562**	**1371.7**	**1120**	12817.3	**5**
M_B	2875.2	3437.9	7286.2	**475.1**	2507.3

Hypothesis 5: Model M'_A will be inferior to M_A as it uses an oversimplified activity location selection.

For the step A2 this is clearly demonstrated in Fig. 5 by $d^A_{ecdf} < d^{A'}_{ecdf}$ for the `leisure` and `shop` activities (only activity types affected by the algorithm selecting activity locations). For the step B2 we get $d^A_{OD} = 3.7 \times 10^{-4} < d^{A'}_{OD} = 4.8 \times 10^{-4}$ which again supports the hypothesised improvement of A over A′.

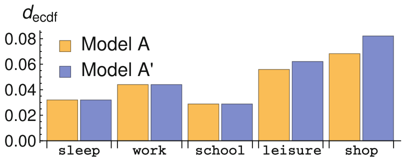


Fig. 5. Activities in space: comparison of Model A to Model A′. M'_A is inferior to M_A for flexible activities ($d^{A}_{ecdf} < d^{A'}_{ecdf}$) based on 18×31 ECDF matrices.

5 Conclusion

We have introduced a detailed methodological framework for data-driven *statistical validation* of multiagent activity-based transport models. The VALFRAM framework compares activity-based models against real-world *travel diaries* and *origin–destination matrices* data. The framework produces several validation metrics quantifying the *temporal*, *spatial* and *structural validity* of activity schedules generated by the model. These metrics can be used to assess the accuracy of the model, guide model development or compare the model accuracy to other models. We have applied VALFRAM to assess and compare the validity of three activity-based transport models of a real-world region comprising around 1 million inhabitants. In the test application, the framework correctly identified strong and weak aspects of each model, which confirmed the viability and usefulness of the framework.

Acknowledgement. This publication was supported by the European social fund within the framework of realizing the project "Support of inter-sectoral mobility and quality enhancement of research teams at Czech Technical University in Prague", CZ.1.07/ 2.3.00/30.0034, period of the project's realization 1.12.2012 – 30.6.2015. This publication was further supported by the Technology Agency of the Czech Republic (grant no. TE01020155).

References

Arentze, T., Timmermans, H.: Albatross: a learning based transportation oriented simulation system. Eirass Eindhoven (2000)
Ben-Akivai, M., Bowman, J.L., Gopinath, D.: Travel demand model system for the information era. Transportation **23**(3), 241–266 (1996)
Bonabeau, E.: Agent-based modeling: methods and techniques for simulating human systems. Proc. Nat. Acad. Sci. U.S.A. **99**(suppl 3), 7280–7287 (2002)
Chen, B., Cheng, H.H.: A review of the applications of agent technology in traffic and transportation systems. IEEE Trans. Intell. Transp. Syst. **11**(2), 485–497 (2010)
de Dios Ortúzar, J., Willumsen, L.: Modelling Transport. Wiley, Hoboken (2011)
Gärling, T., Kwan, M.-P., Golledge, R.G.: Computational-process modelling of household activity scheduling. Transp. Res. Part B Methodol. **28**(5), 355–364 (1994)
Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)

Hollander, M., Wolfe, D.A., Chicken, E.: Nonparametric Statistical Methods, 3rd edn. Wiley, Hoboken (2013)

Jones, P.M., Dix, M.C., Clarke, M.I., Heggie, I.G.: Understanding travel behaviour. Number Monograph (1983)

Jones, P.M., Koppelman, F., Orfeuil, J.-P.: Activity analysis: state-of-the-art and future directions. Developments in dynamic and activity-based approaches to travel analysis, pp. 34–55 (1990)

Klügl, F.: Agent-based simulation engineering. Ph.D. thesis, Habilitation Thesis, University of Würzburg (2009)

Law, A.M.: Simulation Modeling and Analysis, 4th edn. McGraw-Hill, New York (2007)

Law, A.M.: How to build valid and credible simulation models. In: Proceedings of the 2009 Winter Simulation Conference (WSC), pp. 24–33. IEEE (2009)

Manning, C.D.: Foundations of Statistical Natural Language Processing. MIT Press, Cambridge (1999)

Sokal, R.R., Rohlf, F.J.: Biometry: The Principles and Practices of Statistics in Biological Research, 3rd edn. W.H. Freeman, New York (1994)

High Performance Computing for MABS

GPU Environmental Delegation of Agent Perceptions: Application to Reynolds's Boids

Emmanuel Hermellin(✉) and Fabien Michel

LIRMM - CNRS, University of Montpellier, 161 rue Ada, 34095 Montpellier, France
{emmanuel.hermellin,fmichel}@lirmm.fr

Abstract. Using Multi-Agent Based Simulation (MABS), computing resources requirements often limit the extent to which a model could be experimented with. Regarding this issue, some research works propose to use the General-Purpose Computing on Graphics Processing Units (GPGPU) technology. GPGPU allows to use the massively parallel architecture of graphic cards to perform general-purpose computing with huge speedups. Still, GPGPU requires the underlying program to be compliant with the specific architecture of GPU devices, which is very constraining. Especially, it turns out that doing MABS using GPGPU is very challenging because converting Agent Based Models (ABM) accordingly is a very difficult task. In this context, the *GPU Environmental Delegation of Agent Perceptions* principle has been proposed to ease the use of GPGPU for MABS. This principle consists in making a clear separation between the agent behaviors, managed by the CPU, and environmental dynamics, handled by the GPU. For now, this principle has shown good results, but only on one single case study. In this paper, we further trial this principle by testing its feasibility and genericness on a classic ABM, namely Reynolds's boids. To this end, we first review existing boids implementations to then propose our own benchmark model. The paper then shows that applying GPU delegation not only speeds up boids simulations but also produces an ABM which is easy to understand, thanks to a clear separation of concerns.

Keywords: Multi-Agent Based Simulation · Flocking · GPGPU · CUDA

1 Introduction

Because Multi-Agent Based Simulation (MABS) can be composed of many interacting entities, studying their properties using digital simulation may require a lot of computing resources. To deal with this issue, the use of General-Purpose computing on Graphics Processing Units (GPGPU) can drastically speed up simulation runs for a cheap cost [8]. GPGPU relies on using the massively parallel architectures of usual graphic cards to perform general-purpose computing. However, this technology implies a very specific programming scheme which requires advanced GPU (Graphics Processing Unit) programming skills [11].

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 71–86, 2016.
DOI: 10.1007/978-3-319-31447-1_5

Because there are many different MAS (Multi-Agent Systems) models, there is no generic way for implementing MAS using GPGPU. It is not about only changing of programming language. With GPGPU, many concepts which are present in sequential programming are no longer available. Especially, important features of object-oriented languages simply cannot be used (inheritance, composition, etc.). So, it is very difficult to adapt an Agent Based Model (ABM) so that it can be run on the GPU. Considering this issue, hybrid systems represent an attractive solution. Because the execution of the MAS is shared between the Central Processing Unit (CPU) and the GPU, it is thus possible to select only what is going to be translated and executed by the graphics card.

In this paper, we propose to challenge the feasibility and interest of the *GPU Environmental Delegation of Agent Perceptions* (*GPU Delegation* for short) principle which is based on an hybrid approach. This principle consists in identifying and delegating to the environment some of the computations made by the agents. A case study is presented in [9] and shows good results in terms of performances, accessibility and reusability. We propose to trial this principle by using it on a classic ABM, namely Reynolds's boids [14].

In Sect. 2 we review how Reynolds's Boids is implemented in several MABS platforms and then propose our own flocking model in Sect. 3. Section 4 presents the *GPU Environmental Delegation of Agent Perceptions* principle. In Sect. 5, we describe the implementation of our model and how we have applied *GPU Delegation* on it. In Sect. 6, we present and discuss the results of our tests. Finally, we conclude and present perspectives in Sect. 7.

2 Reynolds's Boids

2.1 Original Model Overview

Reynolds was interested in achieving a believable animation of a flock of artificial birds and remarked that it was not possible to use a scripted flock motion. That is why Reynolds proposed an ABM based on local individual rules, namely *boids* [14]. Reynolds's idea was that boids have to be influenced by the others to flock in a coherent manner: "*Boid behavior is dependant not only on internal state but also on external state.*".

Reynolds proposes that each agent is subjected to forces that make it move by taking into account the interactions with the others. So, each entity has to follow **three behavior rules**:

- **R.1** Collision Avoidance: Avoid collisions with nearby flockmates
- **R.2** Flock Centering: Attempt to stay close to nearby flockmates
- **R.3** Velocity matching: Attempt to match velocity with nearby flockmates

Reynolds's boids is recognized as one of the most representative ABM and many agent-based platforms integrate their own boids model.

2.2 Boids in Current MABS Platforms

In this section, we compare several implementations of Reynolds's boids that we can find in popular MABS platforms. Among the related works found, we only introduce models we were able to download and try with an open source code: NetLogo, StarLogo, Gama, Mason and Flame GPU. For each model, we describe how the three rules are implemented (Collision Avoidance (R.1), Flock Centering (R.2), Velocity matching (R.3)).

NetLogo. In NetLogo[1] [17], all the agents move and try to get closer to their peers. If the distance between them and the nearest neighbor is too small, the agent tries to get away (avoid collision (R.1)), otherwise the agent aligns with its neighbors (R.2). However, there is no speed management (R.3): All the agents have the same velocity during the entire simulation.

StarLogo. In StarLogo[2] [13], the agent searches for his closest neighbor. If the distance to his peer is too small, then the agent turns and gets away to avoid collision (R.1). Otherwise, it moves toward him and use his direction. The search for cohesion (R.2) is not explicitly expressed and the velocity of the agents is fixed throughout the simulation (R.3).

Gama. In Gama[3] [3], agents first look for a virtual target to follow (a moving object in the environment that initiates the flocking behavior). Once the agents have a target, they move according to three functions that implement Reynolds's rules: A separation function to avoid collision (R.1), a cohesion function (R.2) and an alignment function for speed and direction (R.3). The model differs from Reynolds's because the agents need a target to actually make the flocking.

MasOn. MasOn[4] [7] uses the computation of several vectors to integrate R.1 and R.2. Each agent computes a motion vector composed of an avoidance vector (this is computed as the sum, over all neighbors, of a vector to get away from the neighbors (R.1)), a cohesion vector (this is computed as the sum, over all live neighbors, of a vector towards the "center of mass" of nearby flockers), a momentum vector (a vector in the direction the flocker went last time), a coherence vector (this is computed as the sum, over all live neighbors, of the direction of other flockers are going (R.2)), and a random vector. The velocity is not managed in this model (R.3).

[1] https://ccl.northwestern.edu/netlogo/.
[2] http://education.mit.edu/starlogo/.
[3] https://code.google.com/p/gama-platform/.
[4] http://cs.gmu.edu/~eclab/projects/mason/.

Table 1. Boids in common MABS platforms

Platform	**Compliance with Reynolds's Model**			**Main characteristics**	**Performances**
	Collision R.1	Cohesion R.2	Velocity R.3		
NetLogo	X	X		R.3 is not implemented: Velocity is fixed throughout the simulation	214 ms (CPU / Logo)
StarLogo	X			A minimalist implementation of behavior rules (only the *collision avoidance* is implemented)	*1000 ms (CPU / Logo)
Gama	X	X	X	Flocking behaviour when agents have a target to follow	375 ms (CPU / GAML)
MasOn	X	X		The rules R.1 and R.2 are reinterpreted into a global vector with addition of random components, no speed management	45 ms (CPU / Java)
Flame GPU	X	X	X	The three rules are explicitly implemented	*82ms (GPU / C,XML)

Flame GPU. Flame GPU[5] [15] is the only GPGPU implementation that we were able to test. In this model, R.1 R.2 and R.3 are actually implemented into three independent functions.

Summary. Table 1 summarizes the implementations of Reynolds's rules, sets out the main features of the models and gives performance informations.

Performances. We evaluate for each model the average computation time in milliseconds for an iteration. The purpose of this evaluation is to give an idea of the possibilities of each implementation. So, we use as common parameter an environment of 512×512 containing 4000 agents. Our test configuration is composed of an Intel i7-4770 processor (Haswell generation, 3.40 GHz) and an Nvidia K4000 graphics card (768 CUDA cores).

It has to be noted that for StarLogo, we observed a computation time higher than a second from 400 simulated agents so that we did not push the tests further. Finally, for Flame GPU, it was not possible to modify the number of agents in the simulation which is of 2048.

3 Reynolds's Boids: Our Model and Implementation

From the previous study, we notice disparities between the various presented models. Indeed, Reynolds's rules allow a large variety of interpretations. For instance, we notice that the speed adaptation rule (R.3) is not always taken into account compared to R.1 and R.2 which are implemented in almost every model (except StarLogo). However, when R.3 is implemented, the collective behavior becomes much more convincing in terms of flocking dynamics and movement believability. Also, in some works, alignment and cohesion behaviors are merged. The models clarifying the difference between this two behaviors offer more interesting movements.

[5] http://www.flamegpu.com/.

The model that we propose takes into account the points observed previously: Overall, when the three rules are explicitly considered, the dynamics and the movement of the agents are more convincing. So, our model integrates R.1, R.2 and R.3 and also follows the KISS (Keep It Simple and Stupid) principle in the aim of creating a minimalist version (with as few parameters as possible). So the model focuses only on the speed and the orientation of the agent[6].

Each entity has a global behavior which consists in moving while adapting its speed and direction. To this end, the proximity with the other agents is first tested and then Reynolds's rules are triggered accordingly. More specifically, every agent first looks in its vicinity. If no agent is present, then it continues to move in the same direction. Otherwise, the agent checks if the neighbors are not too close. Depending on the proximity between entities, agents separate (R.1), align with other entities or create cohesion (R.2). Then agents adapt their speed (R.3), move and restart the process. Figure 1 summarizes the global behavior process.

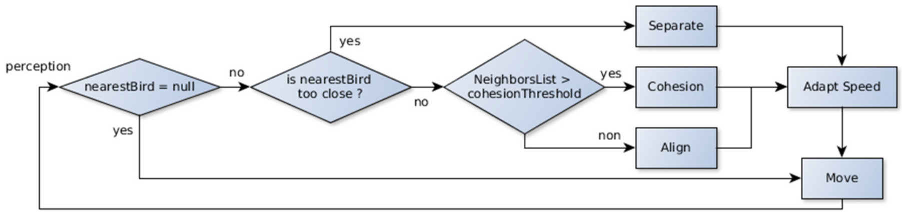


Fig. 1. Flocking: global behavior process

In our model, we have two types of parameters: 5 constants for the model and 3 attributes specific to each agent. The constants are the following ones:

- *fieldOfView* (agent's field of view);
- *minimalSeparationDistance* (minimum distance between agents);
- *cohesionThreshold* (necessary number of agents to begin cohesion);
- *maximumSpeed* (maximum speed of the agent);
- *maximumRotation* (maximum angle of rotation).

The attributes specific to each agent are the following ones:

- *heading* (agent's heading);
- *velocity* (agent's speed);
- *nearestNeighborsList* (the list containing nearest neighbors).

Separation Behavior R.1. When an agent is too close from another one, it separates (R.1). This behavior consists in retrieving the heading of both agents. If these two directions lead to a collision, agent rotates to avoid its neighbor (see Algorithm 1).

[6] The orientation is an angle in degree (between 0 and 360) which gives the heading of the agent according to the landmark fixed in the environment.

Algorithm 1. Separate behavior

```
input  : $myHeading$, $nearestBird$, $maximumRotation$
output: $myHeading$ (the new heading)
1 $collisionHeading \leftarrow$ headingToward($nearestBird$) ;
2 if $myHeading$ inTheInterval($collisionHeading$, $maximumRotation$) then
3     changeHeading($myHeading$);
4 end
5 return $myHeading$
```

Align Behavior R.2. When an agent comes closer to other entities, it tries to align itself with them, by adjusting his direction according to its nearest neighbor (see Algorithm 2).

Algorithm 2. Alignment behavior

```
input  : $myHeading$, $nearestBird$
output: $myHeading$ (the new heading)
1 $nearestBirdHeading \leftarrow$ getHeading($nearestBird$) ;
2 if $myHeading$ isClose($nearestBirdHeading$) then
3     adaptHeading($myHeading$);
4 end
5 else
6     adaptHeading($myHeading$, $maximumRotation$);
7 end
8 return $myHeading$
```

Cohesion Behaviors R.2. When multiple agents are close to each other without having to separate, they have a cohesion behavior. Each agent retrieves the directions of its neighbors and adjusts its own direction based on the average direction found, thus strengthening the cohesion of the group (see Algorithm 3).

Speed Adaptation R.3. Before moving, the agents adapt their speed (R.3). During all the simulation, every agent modifies its speed according to that of its neighbors. If the agent has just executed the behavior of separation (R.1), it accelerates to get free more quickly. Otherwise, the agent adjusts its speed to make it correspond to that of its neighbors (in the limit authorized by the *maximumSpeed* constant).

Algorithm 3. Cohesion behavior

```
input  : $myHeading$, $nearestNeighborsList$
output: $myHeading$ (the new heading)
1  $sumOfHeading, neighborsAverageHeading = 0$ ;
2  foreach $bird$ in $nearestNeighborsList$ do
3      $sumOfHeading += getHeading(bird)$;
4  end
5  $neighborsAverageHeading =$
     $sumOfHeading/sizeOf(nearestNeighborsList)$ ;
6  if $myHeading$ isClose($neighborsAverageHeading$) then
7      adaptHeading($myHeading$);
8  end
9  else
10     adaptHeading($myHeading, maximumRotation$);
11 end
12 return $myHeading$
```

Testing Our Model. We have put online a set of videos that show our model in action[7]. On this page are also available the source codes of the mentioned models and the resources required to test our solution.

4 GPU Environmental Delegation of Agent Perceptions

4.1 MABS and GPGPU

GPGPU Basics. For the purpose of understanding the basics of GPGPU programming, one has to have in mind that it is strongly connected to the underlying hardware architecture of GPU. In this respect, one of the main differences between a CPU and a GPU is the number of processing cores which is far more important in the GPU case.

Today, GPU are composed of hundreds or thousands of processing core (grouped into Streaming Multiprocessors, SM) forming a highly parallel structure able to perform more varied computing. GPGPU relies on using the SIMD (Single Instruction, Multiple Data) parallel model. Also called *stream processing*, the underlying programming approach consists in performing the same operation on multiple data points simultaneously. In other words, GPGPU relies on the simultaneous execution of a series of computations (*kernels*) on a data set (the flow - *stream*).

The related programming models rely on the following work flow: The CPU is called the *host* and plays the role of scheduler. The *host* manages data and triggers *kernels*, which are functions specifically designed to be executed by the GPU, which is called the *device*. The GPU part of the code really differs from

[7] www.lirmm.fr/~hermellin/Website/Reynolds_Boids_With_TurtleKit.html.

sequential code and has to fit the underlying hardware architecture. More precisely, the GPU device is programmed to proceed the parallel execution of the same procedure, the *kernel*, by means of numerous *threads*. These *threads* are organized in *blocks* (the parameters *blockDim.x*, *blockDim.y* characterize the size of these blocks), which are themselves structured in a global grid of blocks. Each *thread* has unique 3D coordinates (*threadIdx.x*, *threadIdx.y*, *threadIdx.z*) that specifies its location within a *block*. Similarly, each *block* also has three spatial coordinates (respectively *blockIdx.x*, *blockIdx.y*, *blockIdx.z*) that localize it in the global *grid*. Figure 2 illustrates this organization for the 2D case. So, each *thread* works with the same *kernel* but uses different data according to its spatial location within the grid[8]. Moreover, each *block* has a limited *thread* capacity according to the hardware in use.

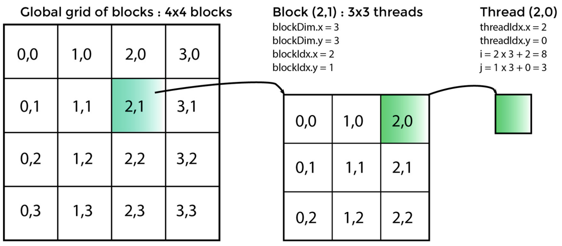


Fig. 2. Thread, blocks, grid organization

So, a multithreaded program is partitioned into blocks of threads that execute independently from each other. The distribution of *blocks* and *threads* on SM may be automatic and is provided by the runtime and drivers. For instance, in the GPGPU platform from Nvidia (Compute Unified Device Architecture, CUDA), a CUDA program can be executed on any number of multiprocessors as illustrated by Fig. 3, and only the runtime system needs to know the physical multiprocessor count.

Implementing MABS Using GPGPU. There are two ways of implementing a model on GPU: (1) *all-in-GPU*, for which the simulation runs entirely on the graphics card and (2) hybrid, the execution of the simulation is shared between the CPU and the GPU. In the first case (1), it is not trivial to take an existing model and translate it to make it work on GPU. GPU are very restrictive in operations and programming and the hardware can only be used in certain ways that requires advanced GPU programming skills. The hybrid approach (2) allows to use jointly the CPU and GPU and thus has two major advantages. Firstly, it brings more flexibility because one can choose what is going to be executed on

[8] In this context, *Thread* is similar to the concept of task: A *thread* may be considered as an instance of the *kernel* which is performed on a restricted portion of the data depending on its location in the global grid (its identifier).

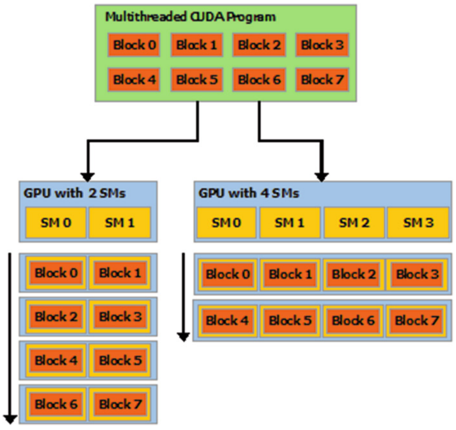


Fig. 3. Automatic scalability (source: Nvidia programming guide)

the GPU, thus providing greater accessibility to the developed tools (as clearly shown in [5,6,16,18]). Secondly, as hybrid systems are modular by design, they make it possible to use agents with complex and heterogeneous architectures. The *GPU Environmental Delegation of Agent Perceptions* principle relies on an hybrid approach.

4.2 Converting Agent Perceptions in Environmental Dynamics

The Principle. *GPU Environmental Delegation of Agent Perceptions* principle was proposed in [9]. This principle consists in making a clear separation between the agent behaviors, managed by the CPU, and environmental dynamics, handled by the GPU. The underlying idea is to identify in the behavior of the agents some computations which can be transformed into environmental dynamics. It has been first stated as follows: *Any agent perception computation not involving the agents state could be translated to an endogeneous dynamic of the environment, and thus considered as a potential GPU environment module.*

Related Works. GPU delegation has to be related to other works which reify parts of the agents' computations in structures related to other concepts such as the interactions or the environment.

In the MABS context, the EASS (*Environment As Active Support for Simulation*) [1] approach aims at strengthening the role of the environment by delegating to it the scheduling policy and adding a filtering system for the perceptions. IODA (*Interaction Oriented Design of Agent simulations*) [4] is centered on the notion of interaction and considers that agent behaviors can be described in a abstract way as a rule called interaction. [12] proposes to reduce the complexity of the models by using an environment-centered approach: Some agent interaction patterns are modeled as environmental dynamics to ease the reusability and the integration of various agent processes.

GPU Delegation on a Case Study. The integration of GPU computations was performed in TurtleKit[9] [10]. TurtleKit is a generic spatial ABM, implemented with Java, wherein agents evolve in a 2D environment discretized in cells. The proposed hybrid approach integrated in TurtleKit focuses on modularity. In this context, this allows to achieve three objectives: (1) maintain accessibility in the agent model while using GPGPU, (2) to scale and work with a large number of agents on large environment sizes and (3) promote re-usability in the particular context of GPU programming.

GPU Delegation has been used only once on a model of multi-level emergence (MLE) [2] of complex structures in TurtleKit. This very simple model relies on a unique behavior which allows to generate complex structures which repeat in a fractale way. The agent behavior is extremely simple and is based on the perception, the spread and the reaction to pheromones. So, in these works, GPU modules dedicated to the perception and the spread of pheromones were proposed.

5 GPU Delegation for Boids

5.1 Applying GPU Delegation

With respect to the underlying hybrid approach, *GPU Delegation* is about identifying specific behaviors which can be turned into environmental dynamics. Especially, *GPU Delegation* states that agent perception computations that do not involve the agents state could be translated into environmental dynamics. For instance, in the previous case study (MLE), the computation related to the agent perceptions used to decide how the agents move according to pheromones (i.e. following gradients) is completely independent from the agents' state: Gradients are equals whatever the agents' states. So a GPU module (environmental dynamics) has been produced for computing pheromones gradients independently from the agents.

In our flocking model, it has not been possible to find a computation which is independent from the agents' attributes. However, we actually identified some computations that can be done *without modifying the agent's state* so that they can be thus translated into an environmental dynamics computed by the GPU.

Indeed, the cohesion behavior consists in averaging the orientations of neighboring agents according to the selected *FieldOfView*[10]. All the agents perform this computation in their own behavior and use the result to adapt their direction. The sequential implementation of this process is as follows:

This loop is very costly when there is a large number of agents because they all perform this computation for each step of the simulation.

[9] http://www.turtlekit.org.

[10] In the context of TurtleKit, FieldOfView is the range which is used to select the cells around the agent.

Algorithm 4. Computing the average of surroundings agents' heading

```
for $bird$ in $nearestNeighborsList$ do
    $sumOfHeading += getHeading(bird);$
end
$neighborsAverageHeading =$
 $sumOfHeading/sizeOf(nearestNeighborsList);$
```

5.2 Designing the Average GPU Module

To achieve GPU translation for the previous computation, we extract information from agents' attributes (heading) and then delegate the associated computation (the loop) to the environment dynamics. To this end, for each simulation step, each agent put its heading value in a 2D array (*headingArray* (a grid matching the size of the environment) according to its position. This array is sent to a GPU module (*average module*) that simultaneously, for each cell within the *FieldOfView*, performs the average of the headings of the surrounding agents. More precisely, each *thread* computes the average for a cell depending on its location in the global GPU grid (its identifiers: i and j in Algorithm 5). The GPU translation thus consists in transforming the sequential computation previously done in the cohesion behavior of the agents into a parallel computation made on the GPU and managed by the environment. Once done, the average headings are available in every cells of the environment. So, the agents can access this data instantaneously with respect to their position (from the 2D array, *flockCentering*, returned by the GPU module) and then adapt their movement accordingly.

Algorithm 5 presents an implementation of the GPU average module: Once the coordinates i and j of the *thread* are initialized, the algorithm tests if the current *thread*'s coordinates do not exceed the size of the environment (represented here by the 2D array *headingArray*). Next, the sum of all the headings are assigned to *sumOfheading*, which is then divided by the number of agents taken into account. The module then returns the array *flockCentering* containing all the averages.

Compared with the sequential version (Algorithm 4, the loop has disappeared. So, one of the main interest of the GPU version lies in the fact that the parallelization of the loop is realized thanks to the hardware architecture, not through code parallelization: Programming with GPU, parts of the code are in the structure.

5.3 Implementation and Integration of the GPU Average Module

The implementation of the GPU average module has been done with CUDA and JCuda[11]. Figure 4 illustrates the integration of the GPU average module in TurtleKit. The implementation was easy thanks to the independence between this module and the agent model.

[11] The JCuda library allows to call GPU kernels, written in C, directly from Java.

Algorithm 5. Average *Kernel*

```
input  : $width$, $height$, $fieldOfView$, $headingArray$ and
         $nearestNeighborsList$
output: $flockCentering$ (the average of directions)
1 $i = blockIdx.x * blockDim.x + threadIdx.x$ ;
2 $j = blockIdx.y * blockDim.y + threadIdx.y$ ;
3 $sumOfHeading, flockCentering = 0$ ;
4 if $i < width$ and $j < height$ then
5     $sumOfHeading$ = getHeading($fieldOfView$, $headingArray[i, j]$);
6 end
7 $flockCentering[i, j] = sumOfHeading/sizeOf(nearestNeighborsList)$ ;
```

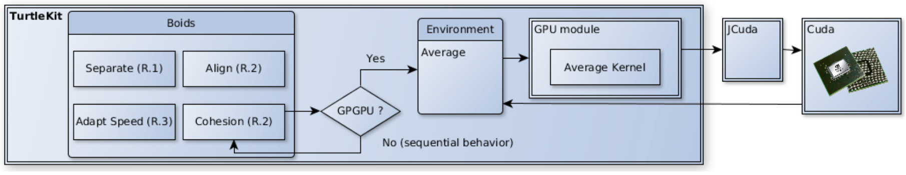


Fig. 4. Integrating the GPU average module in TurtleKit

6 Experimentation

6.1 Experimental Protocol

To trial our implementation of Reynolds's boids and the application of GPU delegation, we simulated several environment sizes while varying the number of agents. We execute successively the sequential version of the model (i.e. the average is computed in the agents' behavior) then the GPGPU version (using the *GPU average module*). To estimate the performances according to the criteria used in Sect. 2, we compare the average computation time in milliseconds for an iteration.

6.2 Performance Tests

For those tests, the configuration is identical as the one used in Sect. 2 and is composed of an Intel i7-4770 processor (Haswell generation, 3.40 GHz) and an Nvidia K4000 graphics card (768 CUDA cores). Figure 5 presents the results obtained for various population sizes in an 256×256 environment (top) and then in an 512×512 environment (bottom).

The use of the GPU module increases the performances by 25 %. However, we notice that the performances is linked to the density of population. Indeed, when the density of the agents in the environment is lower, agents spend fewer time in cohesion and more to align itself and to separate. The density of the agents affects performance of the model when using the GPU module. The tipping point

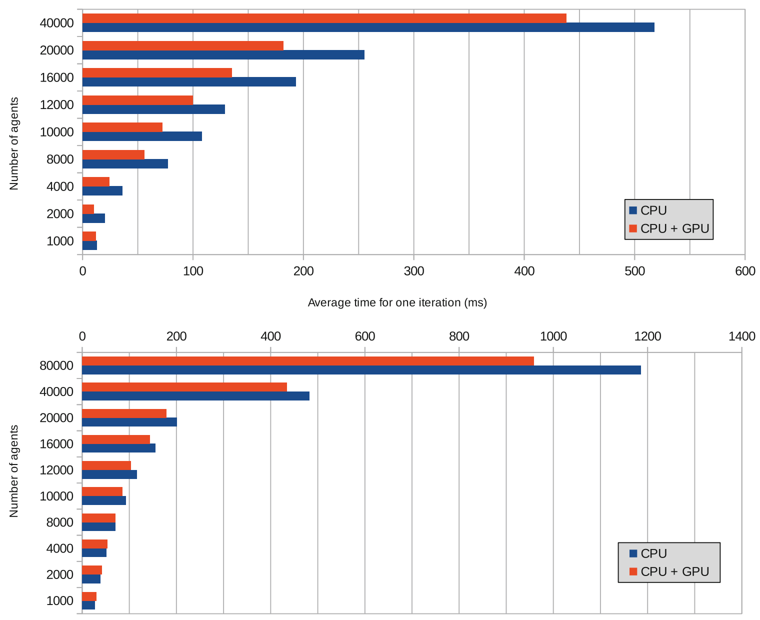


Fig. 5. Comparison of flocking simulations done with and without the GPU. Environment size: 256 (top) and 512 (bottom)

is clearly visible in the results, when the density of present agents exceeds 5 % (respectively 1500 and 8000 entities), the joint use of the CPU and the GPU becomes more effective. So, the more the density of agents in the environment increases, more the observed gains of performances are important.

The performance gains are interesting considering the used hardware: Our Nvidia K4000 embeds 768 CUDA cores while the last Nvidia Tesla K40 card embeds 2880 CUDA cores and the Nvidia Tesla K10 card embeds 3072 cores (two GPU with 1536 cores on the same card). The fast evolution of GPGPU and graphics cards promise very significant gains of performances in the future.

6.3 Discussion

In addition to the observed performance gains, we noticed other benefits related to *GPU Delegation* principle: Translating perception computations done in the agent behavior into environmental dynamics allows to remove a part of the source code and thus simplify the understanding of the behavior. It is more readable because the agent does not have to deal with raw data. Indeed, the agent makes a direct perception in the environment instead of a sequential computation which can be rather heavy.

Another interesting aspect is that the created modules are independent from the considered ABM thanks to this approach. They are thus not limited to the context in which they were defined. That is why we will continue to apply *GPU*

Delegation to create new GPU modules in order to incrementally build a generic GPU modules library. This GPU library will improve the accessibility of the approach and the use of the GPGPU in the MABS context with TurtleKit. This improvement in terms of genericness and accessibility is important because working with GPGPU often leads to implementation difficulties due to the specificity of this technology.

Moreover, GPU delegation is based on a simple criterion which is independent from the implementation. So, GPU delegation allows to convert the model and create the GPU module easily in a rather fast way. TurtleKit being still in alpha release, we are going to continue to work on its architecture in order to make the conversion of a model as simple as possible.

7 Conclusion and Perspectives

In this paper, we described how we used the *GPU Environmental Delegation of Agent Perceptions* principle to implement a classic ABM, namely Reynolds's Boids, using GPGPU. Our purpose was to challenge the genericness and the ease-of-use of *GPU Delegation*. To this end, we needed to evolve GPU delegation so that it can be applied to the boids model. Indeed, find a computation independent from agents' attributes was impossible, so we have identified in the cohesion behavior some computations independent of agent's behaviors. We thus translated these computations into a GPU module and made some tests to see the advantages brought by the *GPU Delegation*.

Our experiments show that, using *GPU Delegation*, it is possible to increase the size of the environments and the number of agents thanks to a speed up which can reach 25 % according to the chosen parameters.

From a software engineering perspective, the use of GPU delegation allows to consider important aspects of MABS with respect to the GPGPU context. By promoting a clear separation between the agent behaviors (handled by the CPU) and environmental dynamics (managed by the GPU), GPU Delegation represents a design guideline which (1) allows to tackle the genericness issue and (2) promotes reusability of created tools. This essential criterion is often neglected in the GPGPU context [9]. GPU delegation allows the creation of generic GPU modules which are independent from the agent models.

Both implementations of the delegation principle, realized with MLE in [9] and flocking here, show that if the analysis of the model is made by keeping in mind the characteristics of the approach, the delegation of the computations and the creation of the GPU module could be very easy and fast, which is a valuable aspect of *GPU Delegation*, especially considering the technical difficulties related with the GPGPU context.

As GPU delegation still requires specific skills, we plan to apply to other models this principle in order to experiencing and continuing to generalize the approach. Then, as a long term perspective, our goal is to propose an explicit design methodology that would provide any MABS end user with a simple and efficient means for addressing scalability issues, without compromising reusability and accessibility which are major issues for the adoption of this technology in the MABS community.

References

1. Badeig, F., Balbo, F., Pinson, S.: A contextual environment approach for multi-agent-based simulation. In: 2nd International Conference on Agents and Artificial Intelligence, ICAART 2010, Spain, pp. 212–217, June 2010
2. Beurier, G., Simonin, O., Ferber, J.: Model and simulation of multi-level emergence. In: ISSPIT 2002, April 2008
3. Grignard, A., Taillandier, P., Gaudou, B., Vo, D.A., Huynh, N.Q., Drogoul, A.: GAMA 1.6: advancing the art of complex agent-based modeling and simulation. In: Boella, G., Elkind, E., Savarimuthu, B.T.R., Dignum, F., Purvis, M.K. (eds.) PRIMA 2013. LNCS, vol. 8291, pp. 117–131. Springer, Heidelberg (2013)
4. Kubera, Y., Mathieu, P., Picault, S.: IODA: an interaction-oriented approach for multi-agent based simulations. Auton. Agent. Multi-Agent Syst. **23**(3), 303–343 (2011)
5. Laville, G., Mazouzi, K., Lang, C., Marilleau, N., Herrmann, B., Philippe, L.: MCMAS: a toolkit to benefit from many-core architecure in agent-based simulation. In: an Mey, D., Alexander, M., Bientinesi, P., Cannataro, M., Clauss, C., Costan, A., Kecskemeti, G., Morin, C., Ricci, L., Sahuquillo, J., Schulz, M., Scarano, V., Scott, S.L., Weidendorfer, J. (eds.) Euro-Par 2013. LNCS, vol. 8374, pp. 544–554. Springer, Heidelberg (2014)
6. Laville, G., Mazouzi, K., Lang, C., Marilleau, N., Philippe, L.: Using GPU for Multi-agent multi-scale simulations. In: Omatu, S., Paz Santana, J.F., González, S.R., Molina, J.M., Bernardos, A.M., Rodríguez, J.M.C. (eds.) Distributed Computing and Artificial Intelligence. AISC, vol. 151, pp. 197–204. Springer, Heidelberg (2012)
7. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.: MASON: a multi-agent simulation environment. Simulation **81**(7), 517–527 (2005)
8. Lysenko, M., D'Souza, R.M.: A framework for megascale agent based model simulations on graphics processing units. J. Artif. Soc. Soc. Simul. **11**(4), 10 (2008)
9. Michel, F.: Translating agent perception computations into environmental processes in multi-agent-based simulations: a means for integrating graphics processing unit programming within usual agent-based simulation platforms. Syst. Res. Behav. Sci. **30**(6), 703–715 (2013)
10. Michel, F., Beurier, G., Ferber, J.: The turtlekit simulation: application to complex systems. In: Workshops Sessions of the Proceedings of the 1st International Conference on Signal-Image Technology and Internet-Based Systems, pp. 122–128. IEEE, November 2005
11. Owens, J.D., Luebke, D., Govindaraju, N., Harris, M., Kruger, J., Lefohn, A.E., Purcell, T.J.: A survey of general-purpose computation on graphics hardware. Comput. Graph. forum **26**(1), 80–113 (2007)
12. Payet, D., Courdier, R., Sébastien, N., Ralambondrainy, T.: Environment as support for simplification, reuse and integration of processes in spatial MAS. In: Proceedings of the 2006 IEEE International Conference on Information Reuse and Integration, USA, pp. 127–131 (2006). IEEE Systems, Man, and Cybernetics Society

13. Resnick, M.: StarLogo: an environment for decentralized modeling and decentralized thinking. In: Conference Companion on Human Factors in Computing Systems, CHI 1996, pp. 11–12. ACM, New York (1996)
14. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. In Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH Computer Graphics 1987, vol. 21, pp. 25–34. ACM, New York (1987)
15. Richmond, P., Walker, D., Coakley, S., Romano, D.M.: High performance cellular level agent-based simulation with FLAME for the GPU. Briefings Bioinform. **11**(3), 334–347 (2010)
16. Sano, Y., Kadon, Y., Fukuta, N.: A performance optimization support framework for gpu-based traffic simulations with negotiating agents. In: Proceedings of the 2014 Seventh International Workshop on Agent-based Complex Automated Negotiations (2014)
17. Sklar, E.: NetLogo, a multi-agent simulation environment. Artif. Life **13**(3), 303–311 (2007)
18. Vigueras, G., Orduña, J.M., Lozano, M.: A GPU-Based multi-agent system for real-time simulations. In: Demazeau, Y., Dignum, F., Corchado, J.M., Pérez, J.B. (eds.) Advances in PAAMS. AISC, vol. 70, pp. 15–24. Springer, Heidelberg (2010)

Parallel Simulations of the Iterated n-Player Prisoner's Dilemma

Diego Queiroz and Jaime Sichman(✉)

Laboratório de Técnicas Inteligentes, PCS
Escola Politécnica da Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158
Cidade Universitária, São Paulo, SP, Brazil
diego.queiroz@usp.br, jaime.sichman@poli.usp.br

Abstract. We present in this work some extended results to those originally published in [8], when we simulated the Iterated n-Player Prisoner's Dilemma in a sequential computer. In [12], we presented a solution to parallel agent-based simulations where agents need to interact with all its neighbours in a von Neumann neighbourhood, aiming to improve the usage of computational resources. By using such parallel techniques, we could better study the effect of several additional parameters of the Iterated n-Player Prisoner's Dilemma simulation, like the grid dimension and the error rate.

1 Introduction

Several computational methods have been applied to social sciences to better understand social phenomena. In particular, the interdisciplinary field of agent-based social simulation [4] allows to model and to simulate how social agents interact and exchange information, and how these processes influence the evolution and behaviour of the population. An essential aspect of this technique is to enable the traceability of micro and macro observations to micro and macro specifications in agent-based models [5].

These simulations do not depend solely on domain dependent parameters, but also on simulation characteristics like the population size [3]. The results of these analysis could conclude that the variation on the population size may have a significant impact on the simulation results [13,14].

One problem that arises in these cases is the high demand of computational power required to simulate large populations. Indeed, there is no way to avoid this problem since the amount of data usually grows significantly according to simulation aspects and population size. A usual solution is to use computer clusters, enabling to distribute the simulation in several independent machines.

In a previous work [12], we presented a solution to better arrange agents in parallel simulations in order to improve the usage of computational resources. That approach intended to be applied on simulations where agents need to interact with all its neighbours in a von Neumann neighbourhood and they are distributed in a grid where cells in the borders are immediately connected to cells in the other side, resembling a ring torus.

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 87–105, 2016.
DOI: 10.1007/978-3-319-31447-1_6

We present in this work some extended results to those originally published in [8], when we simulated the Iterated n-Player Prisoner's Dilemma in a sequential computer. By using the parallel techniques mentioned above, we could better study the effect of several additional parameters of the simulation, like the grid dimension and the error rate.The relation between these approaches are represented in Fig. 1, the one described in this paper corresponds to the right side of the figure. For conciseness, we will not detail here these parallel techniques: the interested reader should refer to [12] for a detailed description.

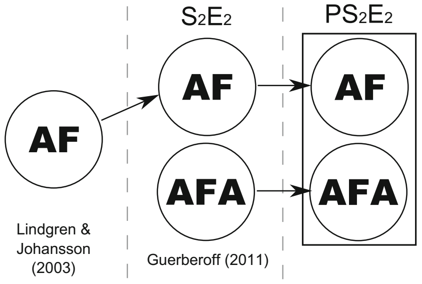


Fig. 1. Approaches to simulate the Iterated n-Player Prisoner's Dilemma

The rest of the paper is organised as follows. In Sect. 2, we introduce very briefly the main concepts of our simulation of the Iterated n-Player Prisoner's Dilemma; interested readers should refer to [8] for more details. The description of the experiments is presented in Sect. 3. The first and second set of experiments, as well as our results analysis, are then discussed in Sects. 4 and 5. Finally, we present our conclusions and further work in Sect. 6.

2 Iterated n-Player Prisoner's Dilemma

The Prisoner's Dilemma (PD) is widely used as a paradigm for the study of the evolution of cooperation in a society of agents. In its iterated version (IPD) with two participants, at each round, agents simultaneously choose one of options: to cooperate ("C") or defect ("D"). If the game is repeated for a sufficient number of periods, there is a chance for the agents to establish mechanisms that support a mutual cooperation. Given the incentives the agents face in this game to defect, it is fundamental that in order to maintain cooperation the free-rider problem must be offset with some punishment mechanism. Moreover, an implementation of IPD requires a *strategy representation language* that enables the agents to represent different strategies, taking into account past interactions.

Axelrod [1] pioneered the use of computer simulation of IPD in an evolutionary environment as a method of studying the dynamics of cooperation. In a tournament of strategies open to the scientific community, he was able to analyse

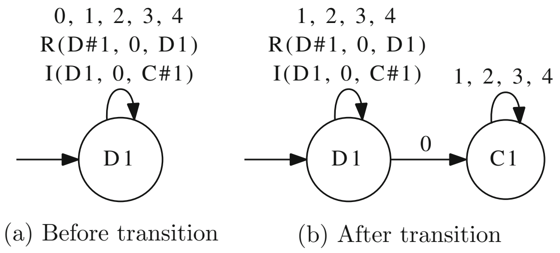


Fig. 2. Adaptive function execution

and compare different configurations of games, types of strategies and other variables, as well as individual and social outcomes. Since then, many other research projects have been conducted modeling agents with heterogeneous strategies playing the IPD in an *evolutionary environment*, like the work described in [6] and [9].

Other works do not feature a pre-determined set of strategies, but study their endogenous development and evolution through the use of genetic and evolutionary mechanisms. Lindgren and Nordahl [11] used genetic algorithms and strategies with changeable memory size in a spatial model where the agents play the Prisoner's Dilemma. Eriksson and Lindgren [7] used finite automata to represent the strategy of agents where the payoff matrices can be randomly modified.

In a previous work [8], we extended the results of [10], where agents' strategies are represented by *finite automata*. In that model, mutations in an evolutionary and spatial environment allow for the endogenous emergence and evolution of strategies. We reproduced the model as specified and extended it to allow the use of *adaptive automata*, which is a richer language than finite automata; it enables the model to execute adaptive actions when transitions between states are fired. Thus, it allows to represent more complex strategies. An example of such adaptive automata is shown in Fig. 2. Supposing that the current state is $D1$ and the input is 0, a transition from $D1$ to $D1$ is fired. However, when this transition is fired the automata adapts to a new configuration, as stated by the two adaptive actions: first, it removes the fired transition ($R(D\#1, 0, D1)$) and then it creates a new transition from $D1$ to a new state labeled $C1$ when the input is 0 ($I(D1, 0, C\#1)$).

The strategies that we used in our simulation, two of them shown in Fig. 3 were the following ones:

- **U strategy:** the agent start playing *defect* and maintain this play, independently of the other players;
- **TT5 strategy:** an approximation of the tit-for-tat strategy for n players, the agent start playing *cooperate* and then follows the last play of the majority of the players[1];

[1] In our case, since we used $n = 5$, the majority of the players is represented by ($n \geq 3$).

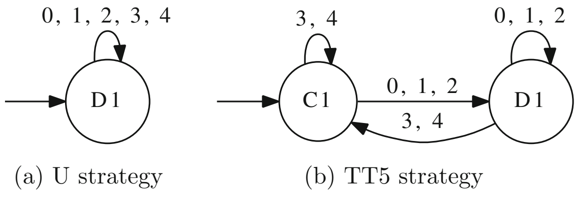


Fig. 3. Strategies used in the simulation

- **CCD strategy:** strategy that contains some adaptive functions (and hence only applicable to adaptive automata), the agent starts playing *cooperate* and maintains this play until another player defects. The agent then calculates c, the number of cooperative rounds. It then starts to play *defect* until all the other agents cooperate again. From this point, the agent decides to play *cooperate* again, but just after some m rounds, this m being a function of the last number of cooperative rounds c.

In the rest of the paper, we will refer to the finite and adaptive automata respectively as B.AF and B.AA models.

3 Description of the Experiments

The main phases of our simulation experiment are shown in Fig. 4. First of all, the agent population is initialized randomly in the grid, each agent having one of the strategies described in Sect. 2. Following their strategy, agents then play 150 times the Iterated n-Player Prisoner's Dilemma with their four von Neumann neighbours. The outcomes of these interactions are inputs to the fitness functions of the evolutionary algorithm. Finally, a last phase involves applying the mutation operator.

3.1 Parameters

We performed several simulations using the parallel techniques described in [12]. These experiments were divided into two sets, detailed in the following sections. Each of them evaluates the impact that the change of some parameters and the amount of processors has on the system outcomes.

To enable the comparison of our results with previous work, we used the same parameters used by [2]:

- k is the number of processors to run the simulation;
- d is the dimension of the lattice;
- g is the number of generations;
- $P\,(\mathrm{m})$ is the probability of mutation, i.e., the probability of changing the strategy after the reproduction phase;
- $P\,(\mathrm{e})$ is the error rate, i.e., the probability that the player does chooses a play not consistent with his strategy.

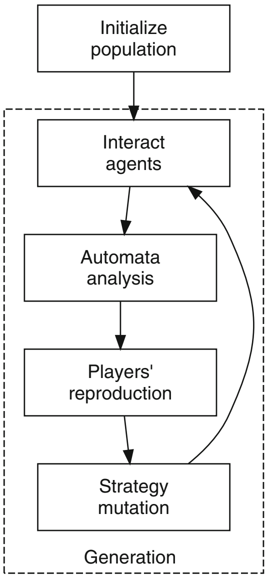


Fig. 4. Simulation phases

Each battery consists of running 32 simulations of both strategy representation models (AF and AFA). The number of simulations has been arbitrarily selected to get sufficient data subject to further statistical analysis. After running the simulations, we collected the run times for each task, as well as the sum of the utilities of all the agents in the lattice.

There are two additional parameters that were not studied in this work: the cost of strategy complexity and the number of iterations per generation. The strategy complexity cost is an aggregated weight to the agent's strategy that benefits those who have simpler strategies, i.e., strategies with fewer states. The number of iterations per generation is the amount of times that agents play the INPPD by making use of their strategy. In this work, these parameters were set at 0,002 and 150, respectively, the same values used in [2].

3.2 Hardware

In order to evaluate the performance obtained with the parallelization of simulation, 5 batteries were run on the Advanced Scientific Computing Laboratory cluster, in the University of São Paulo (USP-LCCA). The cluster used consists of 59 DELL PowerEdge 1950 servers with 2 processors Intel Xeon 5430 (4 cores 2.66 GHz, 12 MB L2 cache and FSB 1 333 MHz), 16 GB DDR2-FBDIMM RAM of 667 MHz and HD SAS 300GB. The cluster uses OS Scientific Linux SL release 5.4 (Boron), 2.6.18–164.11.1.el5_lustre.1.8.3 kernel and uses the TORQUE / PBS system queues manager and scheduler, that ensures the reservation of computational resources so that concurrent processes are not simultaneously executed on the same processor.

Since the cluster resources were reserved, the system could evenly distribute the processors among the different servers according with its availability. Therefore, considering that each server has eight processing cores, simulations using 2 4 or 8 processors are not necessarily performed in a single machine, although there is a likelihood that this has occurred. In other words, a simulation using eight processors could has been performed on a single server, or may have been performed in eight different servers using a single processor in each. Similarly, simulations involving 16 or 32 processors were performed in at least 2 or 4 servers, respectively, since there are no servers with these amounts of processors.

The operational costs generated by the communication between the servers were not considered in this work.

4 First Set of Experiments

A first set of experiments was designed with the unique purpose of evaluating the performance gain achieved by distributing the agents' interactions in the cluster. We made an analysis based on the collection of the simulation run times performed with different amounts of processors, in order to calculate the algorithm's efficiency.

For this purpose, five batteries simulations were performed in each one of the models (B.AF and B.AA). All batteries were composed by 32 runs of 5 000 generations of the game on a 50 × 50 grid, where all players start with the strategy TT5. They interact 150 times per generation, and the strategy complexity cost, mutation rate and error rate were fixed respectively in 0.002, 1 % and 1 %. These values represent the base case test used in this work, identical to the one used by [2], which facilitated the correctness verification of the algorithm during its development.

In this first experiment, five test batteries differed only in the number of processors used in their execution. This number was chosen based on the cluster resources' availability. Simulations were performed with 1, 2, 4, 8, 16 and 32 processors. Table 1 describes the parameters values of these test batteries.

Table 1. Batteries 01–06: Experimental settings

Battery	k	d	g	P (m)	P (e)	Strategy
01	1	50 × 50	5 000	1 %	1 %	TT5
02	2	50 × 50	5 000	1 %	1 %	TT5
03	4	50 × 50	5 000	1 %	1 %	TT5
04	8	50 × 50	5 000	1 %	1 %	TT5
05	16	50 × 50	5 000	1 %	1 %	TT5
06	32	50 × 50	5 000	1 %	1 %	TT5

4.1 Number of Processors

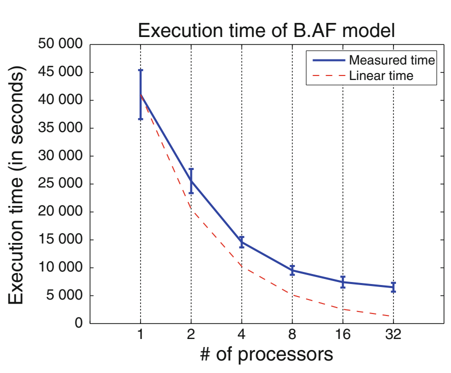
The results obtained with the implementation of the batteries at this stage are listed in Tables 2 and 3 for B.AF and B.AA models respectively. The time displayed corresponds to the absolute real-time simulation of execution (also called real time). Based on the implementation of the Battery 01 each model (corresponding to code execution on a single processor, that is, the sequential code execution), calculated the *speedup* factor and the code *efficiency*. Figure 5 displays the average time of the execution of simulations. The obtained execution times are compared to the ones obtained with a sequential code execution divided by the number of processors used in each experiment (linear time). In the B.AF model, one may notice that the gain obtained by increasing one single processor was around 38 %; this gain increased as more processors were added, reaching an upper limit around 84 %. The same behaviour can also be observed in the B.AA model, with a gain around 52 % and 87 % when using 2 and 32 processors, respectively. Furthermore, when using the B.AAmodel it was observed the occurrence of a *super-linear speedup* when two processors were used. Since the test batteries in the second set of experiments re composed of simulations that share similar features, the results obtained in this first experimental set were used to determine the amount of servers to be used in the subsequent batteries.

Table 2. Batteries 01–06: Average execution time for B.AF model

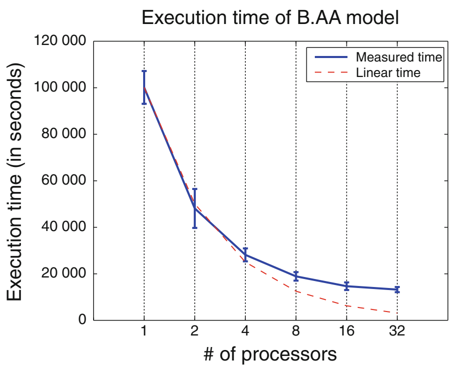
Battery	k	Time		Speedup	Efficiency
		μ	σ		
01	1	11 h 24 min	1 h 13 min	1,0000	1,0000
02	2	7 h 06 min	36 min	1,6063	0,8032
03	4	4 h 03 min	15 min	2,8124	0,7031
04	8	2 h 39 min	13 min	4,3047	0,5381
05	16	2 h 04 min	16 min	5,5233	0,3452
06	32	1 h 49 min	13 min	6,3008	0,1969

Table 3. Batteries 01–06: Average execution time for B.AA model

Battery	k	Time		Speedup	Efficiency
		μ	σ		
01	1	27 h 49 min	1 h 57 min	1,0000	1,0000
02	2	13 h 22 min	2 h 20 min	2,0816	1,0408
03	4	7 h 49 min	46 min	3,5571	0,8893
04	8	5 h 15 min	31 min	5,2977	0,6622
05	16	4 h 05 min	27 min	6,8249	0,4265
06	32	3 h 40 min	19 min	7,5732	0,2367

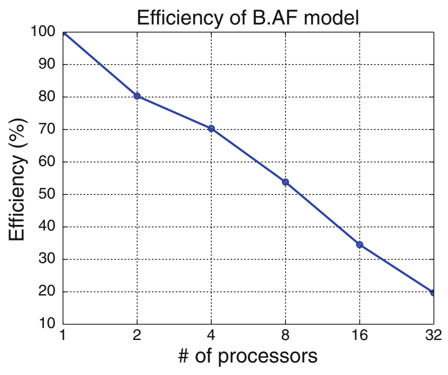


(a) B. AF

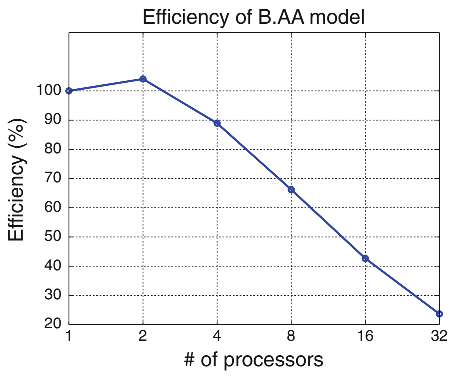


(b) B. AA

Fig. 5. Batteries 01–06: Average execution time for B.AF and B.AA models



(a) B. AF

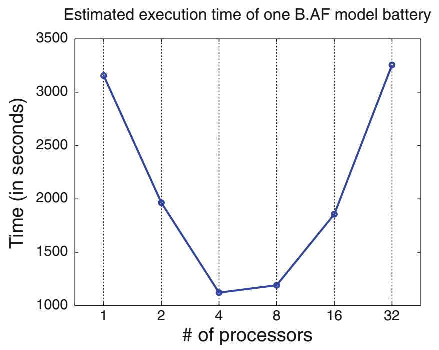


(b) B. AA

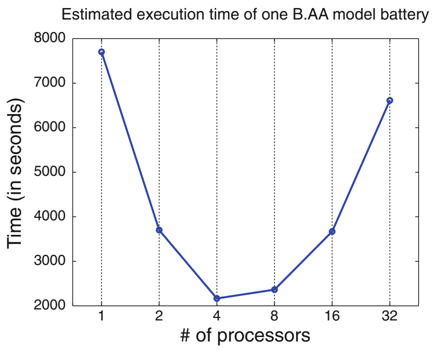
Fig. 6. Batteries 01–06: Efficiency for B.AF e B.AA models

Among the 59 servers in the cluster, 22 are dedicated to the execution of parallel tasks, summing up 176 processors (8 processors per server). As the cluster resources are shared between many users running different tasks, resource allocation is made through a fair-share scheduling, given the server demand, which does not allow a single user to use more than 35 % of all available resources, or 61 processors. Furthermore, the system also allows each user to not hold more than 13 tasks running simultaneously, regardless of amount of resources allocated.

Considering this information, Fig. 7 displays the estimated execution time of each of the models with different number of processors, given the resource allocation in the cluster. This graph shows the total execution time of each battery, composed by the execution of 32 different simulations, considering both the availability and how resources are allocated on the server. In both graphs, it can be seen that the use of *4 processors* results in the shortest time.



(a) B. AF



(b) B. AA

Fig. 7. Execution time estimation for one battery (32 simulations) for B.AF e B.AA models. This value was calculated using the average time of each simulation adjusted by the number of tasks that can be simultaneously processed in the cluster.

Therefore, we have chosen this number of processors to execute the batteries of the next set of experiments.

5 Second Set of Experiments

The second set of experiments aimed to assess how the variation of the simulation parameters influence the outcomes. As in the previous set, all batteries were executed 32 times in each of the proposed models.

In order to assess the gain related with each parameter, the sum of utilities of all agents was collected in each of the generations. Let Va_{ij} be the utility of agent a_{ij} in a particular generation, Eq. 1 calculates the Utility function. This is done by dividing the sum of the utilities of all agents in a generation by the amount of agents in the lattice in order to facilitate the comparison between experiments with different amounts of agents.

$$\text{Utility}\,(\mathbb{U}) = \frac{\sum\limits_{a_{ij}\in\mathbb{U}} V_{a_{ij}}}{|\mathbb{U}|} \tag{1}$$

In this paper, we chose to use Wilcoxon test to accept or to reject our statistical hypothesis, since we didn't suppose that our data was normally distributed. In all experiments the value of the pre-defined significance value α is 0.05, thus rejecting the null hypothesis if p is less than this value. When this happens, we may say that the alternative hypothesis can be statistically confirmed with 95 % of confidence.

5.1 Number of Generations

This first experiment aimed to determine how much the outcomes vary with respect to the system evolution. Differently from other experiment, only a single

Table 4. Battery 07: Experimental settings

Battery	k	d	g	$P\,(\mathrm{m})$	$P\,(\mathrm{e})$	Strategy
07	4	50×50	20,000	1 %	1 %	TT5

battery was run and data was collected every 5 000 generations. The parameters used in this battery are detailed in Table 4. Figure 8 shows the average utility of the set of agents in several generations. We can notice that the experiment involving adaptive finite automata offered the agents an average utility value higher to the one when we used finite automata: the outcomes present a higher average utility of 22.78 %, 18.32 %, 14.96 % and 15.01 % in generations 5 000, 10 000 15 000 and 20 000, respectively. Moreover, we can see that in the B.AA model the system stabilizes rather quickly: after the generation 4 000, it is no longer possible to observe significant changes in the average utility of the agents. However, in the B.AF model, we can observe a subtle growth until generation 16 000.

We can therefore state our first hypothesis:

Hypothesis A: Agent's utility grow in higher generations.

In order to validate this hypothesis, we can express it mathematically as:

$$\text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

where X_1 e X_2 represent a sample of the results obtained in a certain generation. We can then compare each two outcomes to verify the intervals where the hyphotesis holds.

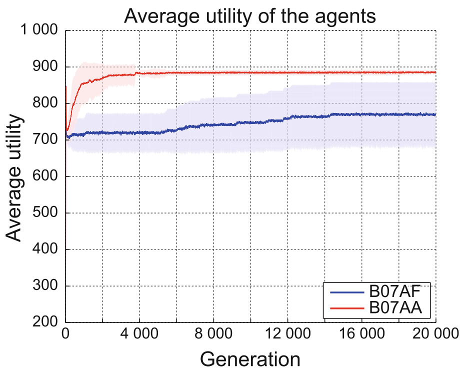


Fig. 8. Battery 07: Average utility obtained by the set of agents

Table 5. Battery 07: Test results for hyphotesis A

X_1	X_2	α	B.AF		B.AA	
			p	H_0	p	H_0
5 000	10 000	0,05	0,049	Rejects	0,402	Doesn't reject
5 000	15 000	0,05	0,009	Rejects	0,165	Doesn't reject
5 000	20 000	0,05	0,007	Rejects	0,089	Doesn't reject
10 000	15 000	0,05	0,042	Rejects	0,089	Doesn't reject
10 000	20 000	0,05	0,102	Doesn't reject	0,105	Doesn't reject
15 000	20 000	0,05	0,445	Doesn't reject	0,452	Doesn't reject

$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

The test results are shown in Table 5. The null hypothesis was not rejected in any test involving the B.AA model, showing that there is no significant gain in this model as generations evolve. However, the test results for the B.AF model showed that there is a significant gain, probably caused by the inherent difficulty that this model has to stabilize. Based on these tests, we decided to fix in the further experiments the number of generations parameter g in 5 000. This choice was made due to two reasons: (i) in this generation the B.AF model is already stabilized; another and (ii) computational constraints given the number of resources available and the number of experiments proposed for this work.

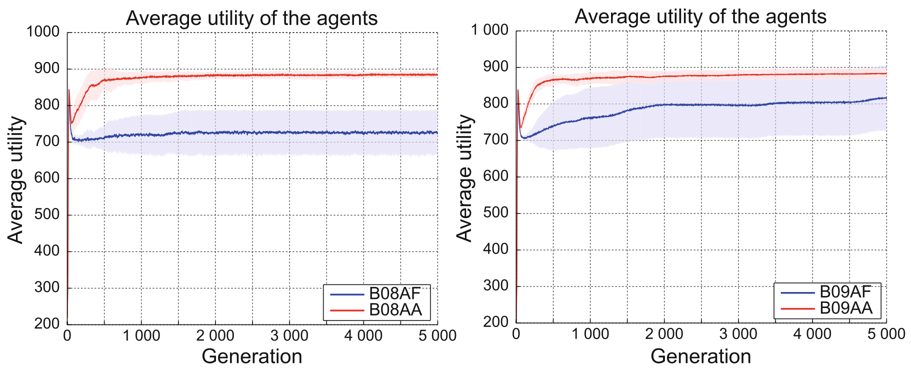
5.2 Grid Size

The following batteries were designed to analyse how the grid dimension influence the strategies used by the agents. We used a 50 × 50 and 128 × 128 dimensions, which were used in [2] and [10], respectively[2]. The parameters settings for these experiments are detailed in Table 6. Figure 9 shown the average utilities obtained by the agents. We couldn't observe a big difference in the average utility when using the B.AA model; however, in the case of the B.AA model, there is a significative increase in the average utility of the agents in the larger lattice, possibly due to the diversity of strategies produced by the experiment, which has facilitated their convergence. We can therefore state our second hypothesis:

Table 6. Batteries 08–09: Experimental settings

Battery	k	d	g	P (m)	P (e)	Strategy
08	4	50 × 50	5 000	1 %	1 %	TT5
09	8	128 × 128	5 000	1 %	1 %	TT5

[2] We decided to increase the number of processors used in battery 9 in order to reduce execution time.



(a) Battery 08 (b) Battery 09

Fig. 9. Batteries 08–09: Average utility obtained by the set of agents

Hypothesis B: Agent's utility grow in grids with higher dimension.

In order to validate this hypothesis, we can express it mathematically as:

$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

The test results are shown in Table 7. As expected, the null hypothesis was not rejected in any test involving the B.AA model. We suppose that this result is inconclusive to fix a good dimensions for the lattice; we intend to make further experiments, for instance using a 50 × 50 grid to compare our results with those obtained by [2]. However, the test results for the B.AF model showed that there is a significant gain for higher dimension grids.

Table 7. Batteries 08–09: Test results for hyphotesis B

X_1	X_2	α	B.AF		B.AA	
			p	H_0	p	H_0
B08	B09	0,05	0,001	Rejects	0,805	Doesn't reject

5.3 Initial Strategy

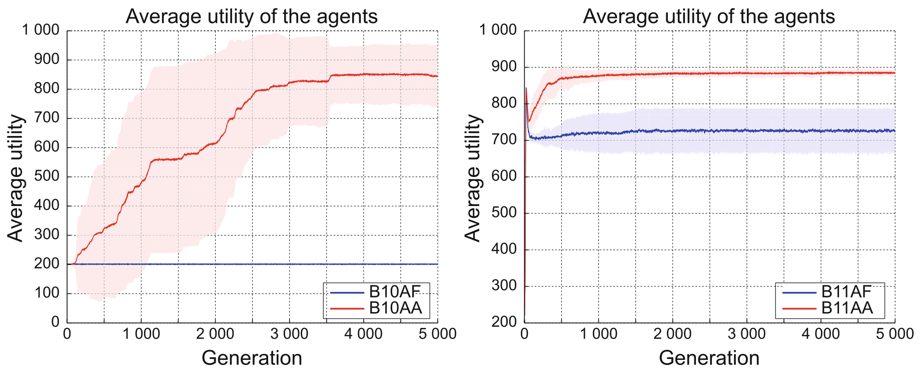
In order to validate the results obtained in [10] and [2], other test batteries were executed where the initial agents strategies were respectively U, TT5 and CCD[3], as presented in Sect. 2. Table 8 describes the parameter settings for

[3] The experiment involving the initial strategy CCD is only applicable to the B.AA model, since it contains transitions with adaptive functions.

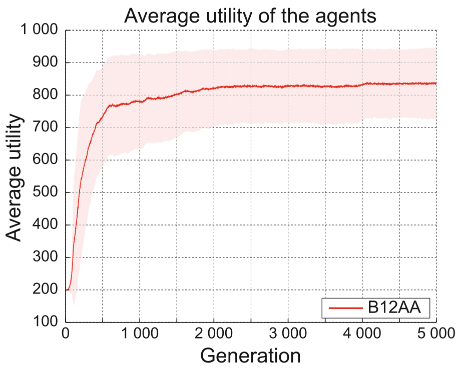
Table 8. Batteries 10–12: Experimental settings

Battery	k	d	g	P (m)	P (e)	Strategy
10	4	50 × 50	5 000	1 %	1 %	U
11	4	50 × 50	5 000	1 %	1 %	TT5
12	4	50 × 50	5 000	1 %	1 %	CCD*

$\star$ Only applicable to the B.AA model.



(a) Battery 10 (b) Battery 11



(c) Battery 12

Fig. 10. Batteries 10–12: Average utility obtained by the set of agents

these experiments. Figure 10 shows the average utility obtained by the agents in the experiments. When using U as initial strategy, the B.AF model was not able to develop a lot of strategies, which prevented the growth of the agents' utilities. However, in the B.AA model, we could observe an increasing in average utility, over the generations. On the other hand, when using CCD as the initial strategy we observed an increasing in the average utility of the agents, without any peaks in the first generations. However, the initial strategy TT5 was the one that has

Table 9. Batteries 10–12: Test results for hyphotesis C

X_1	X_2	α	B.AF		B.AA	
			p	H_0	p	H_0
B10	B11	0,05	$< 0,001$	Rejects	$< 0,001$	Rejects
B10	B12	0,05	N/A	N/A	0,603	Doesn't reject
B11	B12	0,05	N/A	N/A	0,999	Doesn't reject

generated a higher average utility of 5.9 % in generation 5 000. We can therefore state our third hypothesis:

Hypothesis C: A variation in the initial strategy increases the agent's utility.

In order to validate this hypothesis, we can express it mathematically as:

$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

The test results are shown in Table 9. The experiments using TT5 (B11) as the initial strategy had the best performance, and those using U (B10) had the worst performance.

5.4 Mutation Rate

In order to increase the diversity of the population and to observe its effect, some batteries were included varying the probability of mutation $P\,(\text{m})$. Since new strategies are generated only by mutation, this parameter becomes important in our study. The experimental settings of these batteries are described in Table 10, and the results are shown in Fig. 11.

We can therefore state our fourth hypothesis:

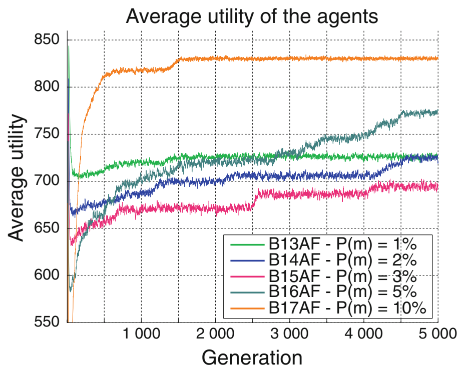
Hypothesis D: If we increase the probability of mutation, the average utility increases.

In order to validate this hypothesis, we can express it mathematically as:

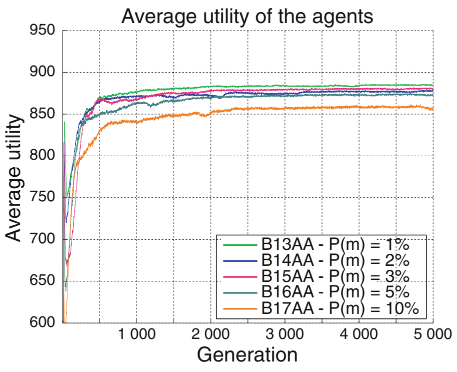
$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

Table 10. Batteries 13–17: Experimental settings

Battery	k	d	g	$P\,(\text{m})$	$P\,(\text{e})$	Strategy
13	4	50×50	5 000	1 %	1 %	TT5
14	4	50×50	5 000	2 %	1 %	TT5
15	4	50×50	5 000	3 %	1 %	TT5
16	4	50×50	5 000	5 %	1 %	TT5
17	4	50×50	5 000	10 %	1 %	TT5



(a) B. AF



(b) B. AA

Fig. 11. Battery 13–17: Average utility obtained by the set of agents

The test results are shown in Table 11. We observe that our hyphotesis was confirmed in the B.AA model; on the other hand, we could not confirm it when using the B.AF model.

Table 11. Batteries 13–17: Test results for hyphotesis D

X_1	X_2	α	B.AF		B.AA	
			p	H_0	p	H_0
B13	B14	0,05	1,000	Doesn't reject	1,000	Doesn't reject
B13	B15	0,05	1,000	Doesn't reject	1,000	Doesn't reject
B13	B16	0,05	0,083	Doesn't reject	1,000	Doesn't reject
B13	B17	0,05	$< 0,001$	Rejects	1,000	Doesn't reject
B14	B15	0,05	0,999	Doesn't reject	0,840	Doesn't reject
B14	B16	0,05	0,450	Doesn't reject	0,999	Doesn't reject
B14	B17	0,05	0,001	Rejects	1,000	Doesn't reject
B15	B16	0,05	0,195	Doesn't reject	1,000	Doesn't reject
B15	B17	0,05	$< 0,001$	Rejects	1,000	Doesn't reject
B16	B17	0,05	0,995	Doesn't reject	1,000	Doesn't reject

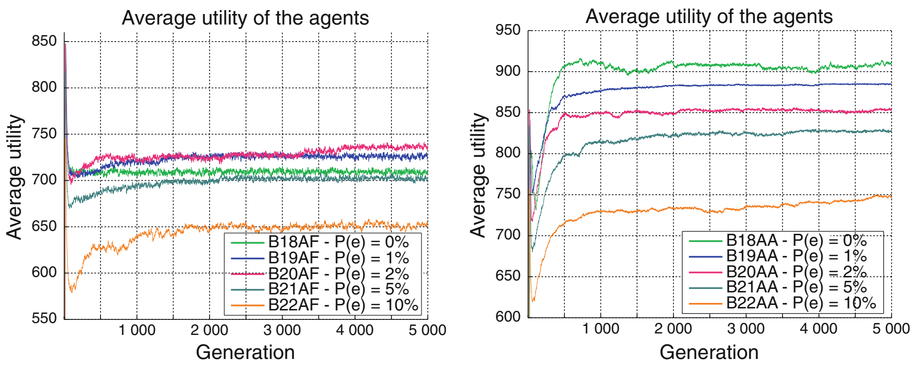
5.5 Error Rate

In these batteries, we evaluated the effect of the error rate. Changing this parameter is especially interesting since it is the only parameter that exerts influence on the agent's reproductive phase. As presented in Table 12, experiments

Table 12. Batteries 18–22: Experimental settings

Battery	k	d	g	P (m)	P (e)	Strategy
18	4	50 × 50	5 000	1 %	0 %	TT5
19	4	50 × 50	5 000	1 %	1 %	TT5
20	4	50 × 50	5 000	1 %	2 %	TT5
21	4	50 × 50	5 000	1 %	5 %	TT5
22	4	50 × 50	5 000	1 %	10 %	TT5

were made adopting probabilities of 0 %, 1 %, 2 %, 5 % and 10 %, where 0 % corresponds to the situation where the agents always plays according to his strategy (no mistakes). Figure 12 displays the agents' average utility in the two models. In the B.AF model, the 2 % value was the one that generated the greatest average utility, which is 12.74 % higher than the lowest average value obtained, with a value of 10 %. Regarding the B.AA model, the agent's average utility was also reduced when the error rate was increased. The higher average utility was obtained with a value of 0 %, which resulted in an outcome 21.81 % higher than the lowest utility average obtained, when the value was 10 %. We can therefore state our fifth hypothesis:



(a) B. AF (b) B. AA

Fig. 12. Batteries 18–22: Average utility obtained by the set of agents

Hypothesis E: If we increase the error rate, the average utility increases.

In order to validate this hypothesis, we can express it mathematically as:

$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

The test results are shown in Table 13. They show rather clearly that the error rate affects negatively the agents' average utility in both models.

5.6 Model Expressivity

Finally, our last test batteries aimed to verify the influence of the complexity of the strategy representation formalism on the simulation outcomes. In order to do so, we use all the results obtained in the second set of experiments, which comprises the batteries 07 to 22. We can therefore state our sixth and last hypothesis:

Hypothesis F: Using a more expressive language to represent agents' strategies (B.AA model) increases the average utility when compared to a less expressive formalism (B.AF model).

In order to validate this hypothesis, we can express it mathematically as:

$$H_0 = \text{Utility}\,(X_1) \geq \text{Utility}\,(X_2)$$
$$H_A = \text{Utility}\,(X_1) < \text{Utility}\,(X_2)$$

The test results are shown in Table 14. Here we can see clearly that the use of the B.AA model to represent the agents' strategies resulted in an increase in the

Table 13. Batteries 18–22: Test results for hyphotesis E

X_1	X_2	α	B.AF		B.AA	
			p	H_0	p	H_0
B18	B19	0,05	0,944	Doesn't reject	1,000	Doesn't reject
B18	B20	0,05	0,921	Doesn't reject	1,000	Doesn't reject
B18	B21	0,05	0,997	Doesn't reject	1,000	Doesn't reject
B18	B22	0,05	0,668	Doesn't reject	1,000	Doesn't reject
B19	B20	0,05	0,628	Doesn't reject	1,000	Doesn't reject
B19	B21	0,05	1,000	Doesn't reject	1,000	Doesn't reject
B19	B22	0,05	0,892	Doesn't reject	1,000	Doesn't reject
B20	B21	0,05	1,000	Doesn't reject	1,000	Doesn't reject
B20	B22	0,05	0,988	Doesn't reject	1,000	Doesn't reject
B21	B22	0,05	0,977	Doesn't reject	1,000	Doesn't reject

Table 14. Batteries 07–22: Test results for hyphotesis F

Battery	α	p	H_0	Battery	α	p	H_0
07	0,05	$<$ 0,001	Rejects	15	0,05	$<$ 0,001	Rejects
08	0,05	$<$ 0,001	Rejects	16	0,05	$<$ 0,001	Rejects
09	0,05	$<$ 0,001	Rejects	17	0,05	$<$ 0,001	Rejects
10	0,05	$<$ 0,001	Rejects	18	0,05	$<$ 0,001	Rejects
11	0,05	$<$ 0,001	Rejects	19	0,05	$<$ 0,001	Rejects
12	N/A	N/A	N/A	20	0,05	$<$ 0,001	Rejects
13	0,05	$<$ 0,001	Rejects	21	0,05	$<$ 0,001	Rejects
14	0,05	$<$ 0,001	Rejects	22	0,05	$<$ 0,001	Rejects

population's utility in all experiments. This can lead us to conclude that the use of more complex mechanisms for the representation of strategies influences positively the gain of society.

6 Conclusions and Further Work

In this paper, we presented new simulation results for the Iterated n-Player Prisoner's Dilemma, thus extending our previous work that was originally published in [8]. These extended results were made possible by running the simulation in a cluster, applying the parallelization techniques that we have presented in [12]. By using such techniques, we could better study the effect of several additional parameters on the simulation outcome, such as the grid dimension and the error rate.

As further work, we intend to better explore and analyse the strategies produced by the evolutionary algorithm, in both B.AF and B.AA models. We could also extend the set of languages to represent the strategies, by applying probabilistic models like Markov models. We also intend to investigate the main characteristics that guarantee a cooperative behaviour of the population.

Acknowledgments. Jaime Sichman is partially supported by CNPq, Brazil, grant # 303950/2013-7. During this work, Diego Queiroz was supported by CNPq, Brazil; now, he is currently supported by Capes, Brazil. We also thank the CCE-USP, and more particularly the LCCA-Laboratory of Advanced Scientific Computation of the University of São Paulo, whose parallel machines were used in our experiments.

References

1. Axelrod, R.: The Evolution of Cooperation. Basic Books, New York (1985)
2. Bó, I.G.L.: Influência da complexidade da representação de estratégias em modelos evolucionários para o Dilema do Prisioneiro com n jogadores. Ph.D. thesis, Universidade deSão Paulo, São Paulo (2008). http://www.teses.usp.br/teses/disponiveis/3/3141/tde-31032008-161326/?&lang=pt-br
3. Bosse, T., Gerritsen, C., Hoogendoorn, M., Jaffry, W., Treur, J.: Agent-based and population based simulations of displacement of crime. In: Proceedings of the Seventh IEEE/WIC/ACM International Conference on Intelligent Agent Technology. IAT 2008, pp. 469–476, Sydney (2008)
4. David, N., Marietto, M.B., Sichman, J., Coelho, H.: The structure and logic of interdisciplinary research in agent-based social simulation. J. Artif. Soc. Soc. Simul. 7(3) (2004)
5. David, N., Sichman, J.S., Coelho, H.: Towards an emergence-driven software process for agent-based simulation. In: Sichman, J.S., Bousquet, F., Davidsson, P. (eds.) MABS 2002. LNCS (LNAI), vol. 2581, pp. 89–104. Springer, Heidelberg (2003)
6. Delahaye, J., Mathieu, P.: Complex strategies in the iterated prisoner's dilemma. In: Proceedings of the 1994 Chaos & Society Conference, pp. 283–292. IOS Press (1994)

7. Eriksson, A., Lindgren, K.: Cooperation driven by mutations in multi-person Prisoner's Dilemma. J. Theor. Biol. **232**(3), 399–409 (2005)
8. Guerberoff, I., Queiroz, D., S. Sichman, J.: Studies on the effect of the expressiveness of two strategy representation languages for the iterated n-player prisoner's dilemma. Revue d'intelligence artificielle 25(1), 69–82, February 2011. http://ria.revuesonline.com/article.jsp?articleId=15967
9. Ifti, M., Killingback, T., Doebeli, M.: Effects of neighbourhood size and connectivity on spatial Continuous Prisoner's Dilemma. Arxiv preprint q-bio.PE/0405018 (2004)
10. Lindgren, K., Johansson, J.: Co-evolution of strategies in n-person Prisoner'sDilemma. In: Crutchfield, J.P., Schuster, P. (eds.) Evolutionary Dynamics: Exploring the Interplay of Selection, Neutrality, Accident, and Function, pp. 341–360. Oxford University Press, New York (2003). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.123.7042
11. Lindgren, K., Nordahl, M.G.: Evolutionary dynamics of spatial games. Physica D: Nonlinear Phenom. **75**(1–3), 292–309 (1994)
12. Macedo, D.D.Q., Sichman, J.S.A.: Analysis of von neumann neighborhoods in parallel multi-agent simulations. In: 2010 Second Brazilian Workshop on Social Simulation. pp. 27–32. IEEE, October 2010. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6030010
13. Németh, A., Takács, K.: The evolution of altruism in spatiallystructured populations. J. Artif. Soc. Soc. Simul. **10**(3), 4 (2007). http://jasss.soc.surrey.ac.uk/10/3/4.html
14. Walsh, W.E., Das, R., Tesauro, G., Kephart, J.O.: Analyzing complex strategic interactions in multi-agent systems. In: Game Theory & Decision Theory Workshop, AAAI (2002)

MABS Applications

Agent-Based Visualization: A Simulation Tool for the Analysis of River Morphosedimentary Adjustments

Arnaud Grignard[1](✉), Guillaume Fantino[2], J. Wesley Lauer[3], Alexandre Verpeaux[1], and Alexis Drogoul[1]

[1] IRD, UMI 209 UMMISCO, IRD France Nord, 93143 Bondy, France
agrignard@gmail.com
[2] Geopeka, Lyon, France
[3] Civil and Environmental Engineering, Seattle University, Washington, USA

Abstract. Spatially explicit agent-based models and simulations are playing an increasing role in the modelling of complex natural and social systems. The ARCHEM project belongs to this new research area. It proposes a new methodology to visualize the fine-scale sediment transport of a river. In this paper, we present the first implementation of ARCHEM on a case study of the Rhone river. Even though visualization cannot replace the analysis of simulation results, it often constitutes a more accessible medium that can facilitate more specific and accurate interpretations of simulation output. It has the advantage of offering immediate feedback as well as a way to interact with and analyze results. We show how to support multiple viewpoints and different levels of abstraction using an agent-based visualization approach. We present a specific application focusing on dynamical 3D rendering of a GIS file and the analysis of morphosedimentary adjustments.

Keywords: Agent-based model · Visualization · 3D · GIS · Human-environments interactions · Sediment deposition

1 Introduction

In complex systems modelling, which by definition requires a mixture of different entities at different levels of organization, visualization of structures emerging as a result of the interactions among various system components is one of the hardest challenges for research in information technology [2]. One of the difficulties is to provide generic tools to easily define represent, abstract and interact with dynamical structures. Recent academic research has failed in this area, often relying on ad-hoc approaches which are difficult, if not impossible, to change and to reuse in other models [1]. There is still a lack of generic integrated analysis and visualization tools running online [10]. To overcome this limitation, our work presents solutions and tools that enable information visualization using an agent-based approach implemented in the dedicated platform GAMA [5].

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 109–120, 2016.
DOI: 10.1007/978-3-319-31447-1_7

This framework supports the definition of flexible, adaptable and reusable visualization components using abstraction and refinement concepts such as clustering, spatio-temporal aggregation and multi-layer representation.

Our project enhances dynamic information visualization by multiplying the point of view and defining different levels of abstraction. It enables (i) to define different points of view using observer (camera) position and agent aspect (ii) to build abstraction and generalization with dedicated macro-agents used to aggregate a set of micro agents or representing more abstract information coming from data-mining tools (iii) to use those abstraction to control the reference model. This practice enables answers from multiple perspectives, each with different requirements wherein different users with different skills work on the same model, in multi-level modeling, to represent the different levels involved and also in co-modeling to facilitate the coupling between heterogeneous models.

The ARCHEM project has the goal of presenting tools that can improve management decisions regarding sediment recharge operations in a river system whose sources of sediment are limited. These tools are intended to assess the spatial and temporal impact (travel speed, particle size and channel geometry) of these operations based on several scenarios for sediment re-injection. In this paper we present the first phase of this study wherein we focus on visualizing simulations for the sediment recharge scenarios. The results of these simulations are dynamically displayed in the framework of a GIS mapping system.

The paper is organized as follows: Sect. 2 presents the context and the related work. Section 3 presents the methodology used which consists of using an agent-based model dedicated to information visualization that we call agent-based visualization. Section 4 presents the implementation of our approach in five different steps. Section 5 discusses the results and the future work. Finally Sect. 6 concludes the paper.

2 Context and Related Work

The Rhone Valley Human-Environment Observatories (OHM) is responsible for managing the riverscape that extends from Geneva to the Mediterranean, including the main channel and all surrounding water bodies and riparian areas, some of which are prone to flooding. The study area is influenced by numerous alterations all along the river course as shown in Fig. 1. These are the result of two key factors: construction of the navigable channel (1840–1910) and the installation of hydroelectric facilities (1948–1986). These determine the dynamics of the contemporary landscape and influence the distribution of human activities.

2.1 The Rhone River: A Century of Human Planning

River management along the Rhone, mainly related to navigation, has profoundly changed the channel geometry in the twentieth century (incision of main channel, less frequent flooding of channel margins) and resulted in degradation of benthic habitats. Recent research, conducted as part of the OSR (master plan for the restoration of river dynamics margins of the Rhone), showed the potential value of demolition of some of these old facilities in Vieux-Rhone for restoration of the river bed by re-expansion [9]. This work will most likely lead to sediment recharge operations. Decision making around this type of operation is difficult because the expected environmental gains are difficult and complex to evaluate and represent. Therefore, the development of modeling tools and geovisualisation to inform the debate around these issues represents an important response to both scientific and operational issues.

The theoretical approach to the hydro-sedimentary operation [6] is now possible to finely model scale sediment transport of a section. This type of theoretical approach has recently been used to model the sedimentary processes on a 40 km river reach downstream from a hydroelectric project (lower valley of the Ain).

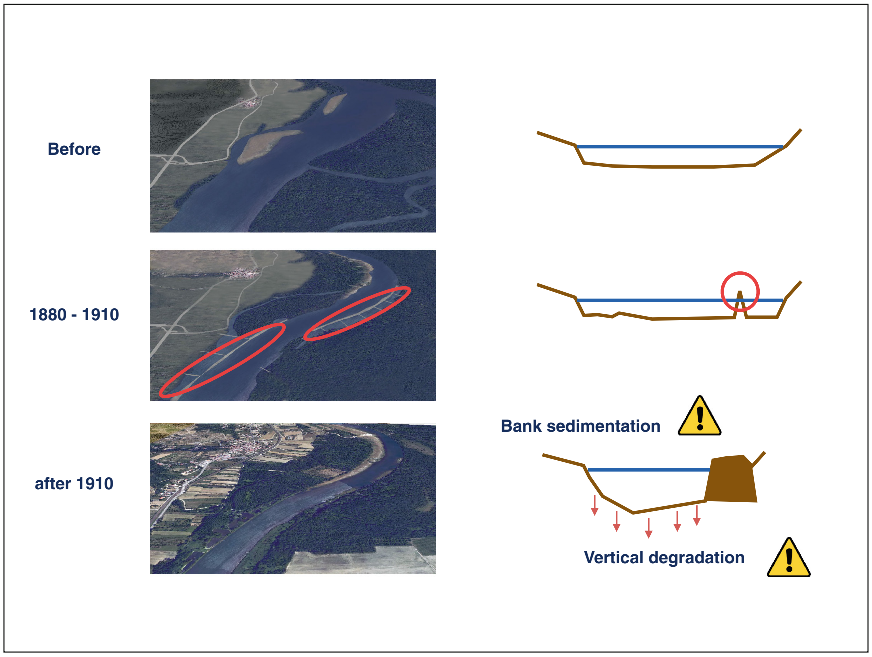


Fig. 1. The Rhone River has been modify to facilitate fluvial transport with the creation of dykes used to focus the flow. These changes have caused bank sedimentation and a vertical degradation of the river.

The Rhone is also sufficiently well documented in terms of available data to build a model representation of an Old Rhone. It is therefore possible to use this type of model to represent the effect of recharge operations.

2.2 Tools for Modelling Human-Environment Interactions

Modeling complex systems such as Human-Environment Interactions requires the utilization of concepts and methods for studying the landscape at different spatiotemporal scales. Among these techniques, agent-based modeling is used to study a system by modeling the entities that comprise the form of agents whose interactions allow the emergence of global dynamics.

Decision making around this type of operation is delicate, and development of modeling tools and geovisualisation therefore responds to both scientific and operational issues. However communication about results of these models can still to be improved if it is to be used for decision support.

GIS. Current GIS software manages large GIS datasets and can perform complex spatial analysis on them, but it is important to consider the following two points. (i) No existing packages manages the 3D natively. While it is a safe bet that in a few years 3D will be natively integrated into the GIS data processing, our approach is one of the most viable solutions to display GIS data in 3D today. (ii) No existing packages can give behavior to GIS objects. Our approach, consisting in reifying those GIS objects, makes it easy to give behavior to GIS data.

ABM. Public research and development investments gave birth to many ABMs software environments. ABMs software environments can be divided into three main categories. The first one enables the user to define a model using generic languages such as Java, C++ or Python and dedicated libraries. Those platforms are hard to use for non-computer scientist but are suitable for large scale and complex models with many agents and processes involved (Repast [8] and Swarm [13]). The second category encompasses platforms that enable the definition of the model through a dedicated language like Netlogo [14] or GAMA [5]. They are simpler to use but can have limitations when dealing with large models. The last category is the one where the user can define the model using a graphical language. These platforms do not require any programming skills but are still limited, and include StarLogo [12] or AgentSheets [11]. Using agent-based model for visualization purposes is a relatively new approach, and while most of the existing ABM platforms include visualization tools, only agents present in the reference model are displayed. They do not enable the use of graphical agent for visualization purposes.

3 Methodology and Tools

3.1 Agent-Based Visualization

Agent-based visualization (ABV) consists in using an agent-based model for visualization purposes. As shown in Fig. 2, an agent-based model is a system composed of different *localized* entities (**agent**) evolving in an *environment*. An agent has *attributes*, *behaviour* and abilities of *perception* and *communication* and can *interact* together. Agent-based visualization uses discrete entities responsible of visualization tasks and that can interact with other visual agents. These agents have the possibility of learning about the collective/emergent patterns or behaviors present in the data they handle and to adapt their individual or collective aspect.

The different representations of those initial objects depend on two parameters: the observed attribute and the behavior of each graphical agent. Agent **attributes** (or properties) are the variables manipulated by the agent. They represent the inner characteristics of an agent and can be atomic (int, real, boolean) or more complex (list, matrix, agent). Each agent has a set of internal **behaviours** to modify its attributes (e.g. change my location, change my color, change my size). An agent can have different **aspects** that can be used in different cases depending on the properties the user wants to emphasize. It has a given shape, size, color, etc.

ABV has proven, on various visualization tasks ([3,4]), to be flexible (in terms of defining different graphical representations and viewpoints on a particular dataset), modular (as agents can be added, removed or changed dynamically) and adaptive (as agents can reorganize to dynamically handle new data sources).

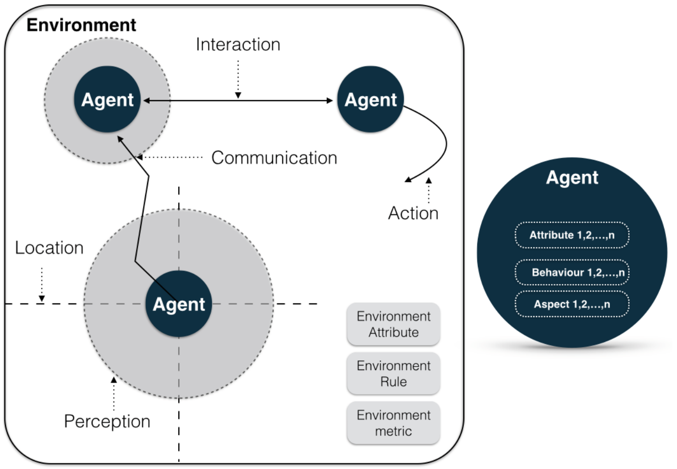


Fig. 2. Agent-based meta-model

4 Applications: Morphosedimentary Adjustements Visualization

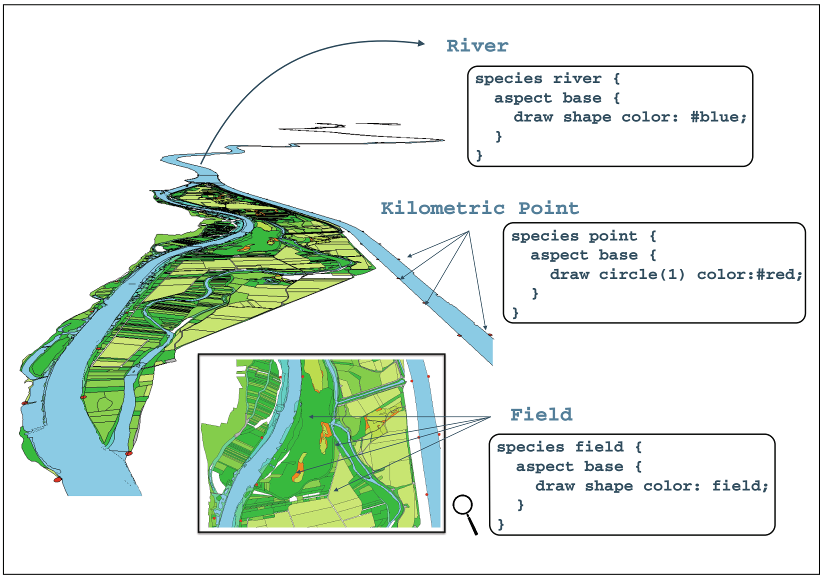
4.1 General Methodology

The goal of the ARCHEM project is to visualize morphosedimentary adjustements. In order to achieve this goal, we use a methodology, implemented in the GAMA platform and using GAML language, consisting of 5 steps:

1. Build an immersive static 3D representation of the watershed.
2. Reconstruct missing data.
3. Build an immersive dynamic 3D representation of the watershed.
4. Animate the watershed.
5. Couple watershed representation with a morphodynamic model.

4.2 Step1: Static 3D Representation

This visualization model consists in converting GIS data into agents and representing them in a 3D space as shown in Fig. 3. This model represents three types of data corresponding to three distinct species of agents on the section of the river studied (the Rhone). The *river* is represented according to its corresponding shapefile. The type of land around the river(*field*) is represented with a specific color that categorizes the different types of area. Finally *kilometric points* located every 500 m are represented by red circles.



```
species river {
  aspect base {
    draw shape color: #blue;
  }
}
```

```
species point {
  aspect base {
    draw circle(1) color:#red;
  }
}
```

```
species field {
  aspect base {
    draw shape color: field;
  }
}
```

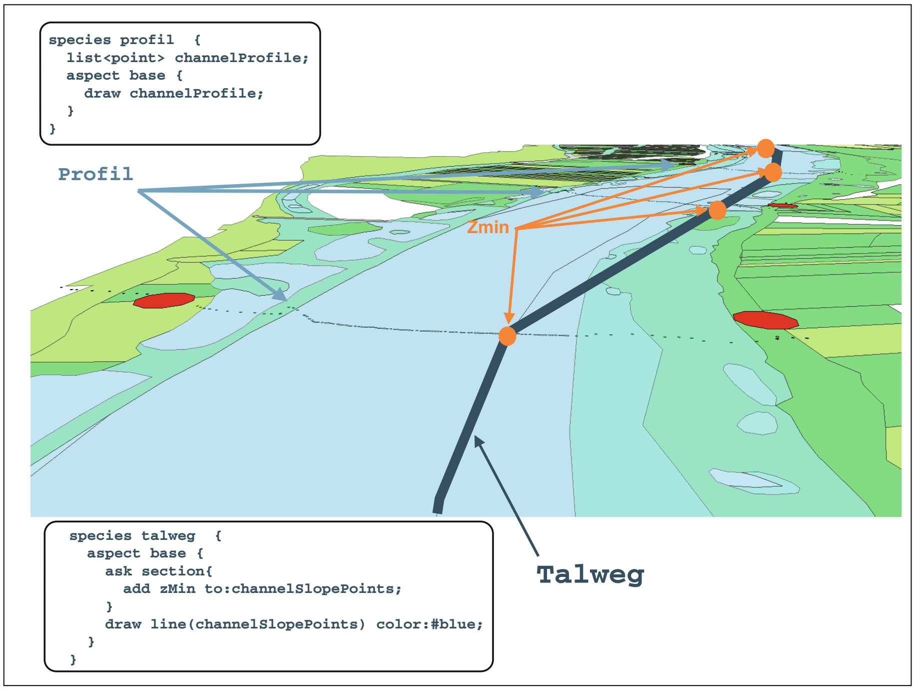
Fig. 3. Static 3D representation and the corresponding GAML code (Color figure online).

4.3 Step 2: Data Reconstruction. Indicator Creation

An agent-based visualization model can adapt itself automatically to the data and create local and global indicators of local and global basis. Unlike conventional data visualization, graphical agents play a big role in the quality of the information displayed thanks to their autonomy and spatially explicit behavior. Information that is not present in the initial dataset can be calculated using the dedicated graphical agents present in the model. The adaptive capacities of agents are used to create new agents during the simulation.

In our case, from the initial agent *kilometric point*, we determine the point with the lowest altitude in the transverse profile in order to build and represent the thalweg (line joining the lowest points of a valley). The thalweg becomes a new agent created during the simulation.

Figure 4 represents a section of the river, on which are represented the transverse profiles and thalweg. The thalweg is a new species with a list of items containing the deepest points of each section and a line connecting all the points of the previous list.



```
species profil  {
  list<point> channelProfile;
  aspect base {
    draw channelProfile;
  }
}
```

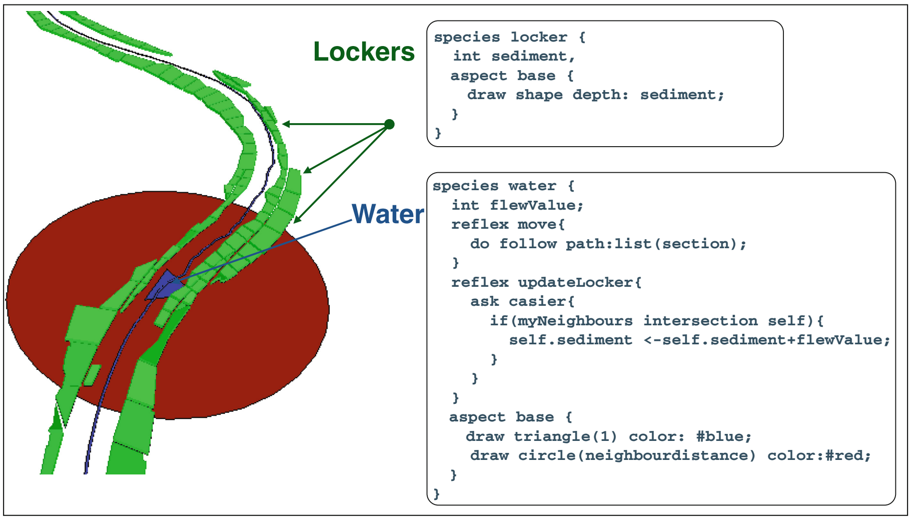
```
species talweg  {
  aspect base {
    ask section{
      add zMin to:channelSlopePoints;
    }
    draw line(channelSlopePoints) color:#blue;
  }
}
```

Fig. 4. Representation of transverse profiles and talweg reconstruction and the corresponding GAML code.

4.4 Step 3: Dynamic 3D Representation

This model illustrates the flexibility and adaptability of our approach by providing a dynamic representation of the sedimentation process. Rainfall data are represented as an agent and coupled with the previous model to visualize its impact on the river.

Distributed along the river are dykes, that is, low areas between rock groins that were themselves built to focus the river into a narrow zone, thereby enabling navigation. These records contain a lot of sediment whose height varies according to rainfall data. The species *water* representing the flow of water illustrates the influence of water flow on the locks. Each agent *water* travels over the course of the river on a given path (in this case the route of the river) and interacts with locks present in its neighbourhood. All locks intersecting this zone are considered part of the neighborhood. The agent *water* will have an effect on the locks (increase or decrease its depth) depending on the speed of the stream at a time *t*. The amount of sediment deposited in the bins is then represented by the height of the lockers as shown in Fig. 5.



```
species locker {
  int sediment,
  aspect base {
    draw shape depth: sediment;
  }
}
```

```
species water {
  int flewValue;
  reflex move{
    do follow path:list(section);
  }
  reflex updateLocker{
    ask casier{
      if(myNeighbours intersection self){
        self.sediment <-self.sediment+flewValue;
      }
    }
  }
  aspect base {
    draw triangle(1) color: #blue;
    draw circle(neighbourdistance) color:#red;
  }
}
```

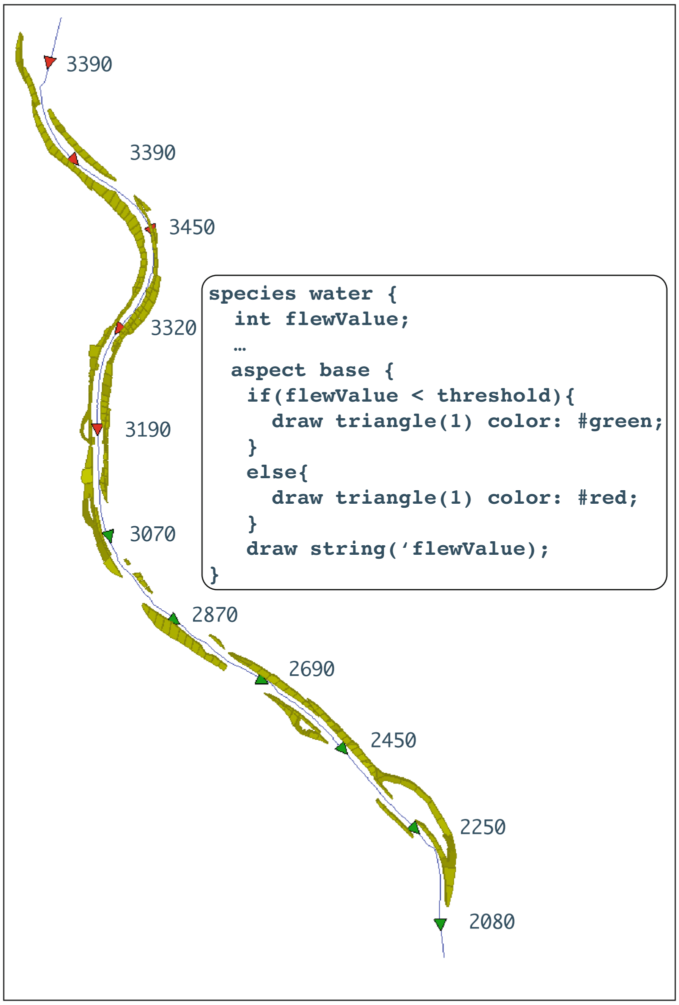
Fig. 5. Modelling of water flow and the corresponding GAML code. The agent *water* moves along the river and updates the height lockers located in its neighbourhood to provide an animated representation of the sedimentation process.

4.5 Step 4: Coupling with Real Data

Historical records of flow in the river can be used to replicate past events. Data are stored in a file where each line corresponds to the daily flow measurement reading, in m^3/s. At each iteration, depending on the value of the stream flow, an agent *water* will be created. For example the first value in the file is 3070 m^3/s, this corresponds to the creation of an agent *water* type *fast* represented by a green triangle and causing a reduction in the height of the sediment in its passage. The first 13 iterations of the simulation are shown in Fig. 6 below.

4.6 Step 5: Coupling with External Hydro-Sedimentary Model

In previous models, the water flow was represented by agents having a direct influence on the river. This approach, however, remains an approximate method to represent the phenomenon instead of actually model it. The hydro-sedimentary models now allow accurate modeling scale sediment transport in a river stretch.



```
species water {
  int flewValue;
  …
  aspect base {
   if(flewValue < threshold){
     draw triangle(1) color: #green;
   }
   else{
     draw triangle(1) color: #red;
   }
   draw string(‘flewValue);
}
```

Fig. 6. Creating *water* agents from rainfall data and the corresponding GAML code.

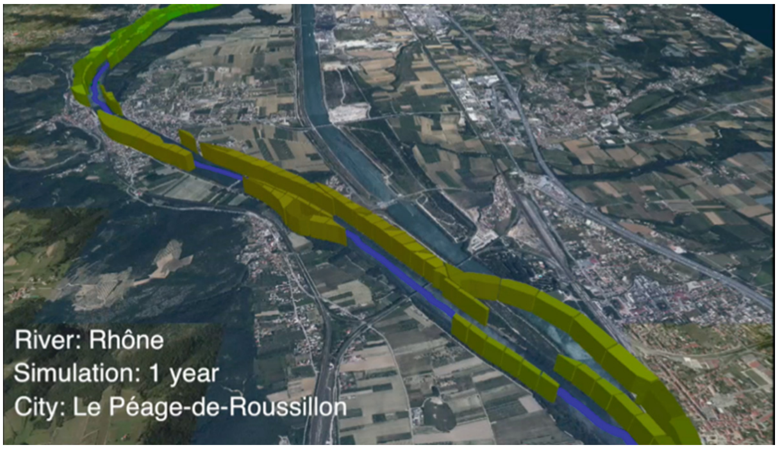


Fig. 7. Spatialized representation of the MAST-1D model.

Morphological changes in the river were modeled using a hydraulic/morpho-dynamic model (Mast-1D [7]). However this kind of model only gives numerical values corresponding to the water depth for each section. These numerical values allow a detailed analysis of the behavior of the river but lack of intelligibility.

We spatially represent the outputs of the MAST-1D model. We provide a generic representation of the model and then propose a spatial representation in which the outputs will be represented on GIS corresponding to the river studied. As shown in Fig. 7, this model represents for each section, the channel (*Chanel*), the thickness of the active band (*Active Layer*), the thickness of the floodplain (*Floodplain*) and finally the thickness of the outer floodplain (*Distal Food plain*) which are output values coming from the MAST-1D model.

In Fig. 7, the left part is a generic representation of the model (MAST-1D). This non-spatial model provides values of various parameters every 500 m along the channel. The various parameters useful for spatial representation of the model are updated at each iteration of the simulation. The changing course of the river is then represented dynamically by reading the values of parameters outputed by the MAST-1D model and updating the look of each agent corresponding to a section of the river and at each iteration.

For a more realistic rendering, each agent representing a section has a position so it is possible to apply a translation to a specific position to apply the model to any river. The Fig. 7 is a representation of the MAST-1D model on the Rhone river.

5 Results and Discussions

The first results of this approach are very encouraging[1]. Such coupling has helped to build a space-time representation of the morphodynamics of the

[1] An animated result can be found here http://youtu.be/HWctj1ni5Qk.

short-circuited section of the Rhone both retrospectively and prospectively. We are able to represent in an immersive 3D environment a complete watershed with the river and the surrounding fields. Moreover, the dynamical process of sedimentation has been implemented both with synthetic values and with data coming from historical datasets. Finally, the spatialization of a numerical hydro-sedimentary model has been achieved. To the best of our knowledge, this is the first time that such a visualization has been developed.

Future work will consist in analyzing the Rhone morphological adjustment resulting from shrinkage of the bed Girardon developments of the late 19th century and evaluating the effect of changes in flood regime following the construction of a bypass channel in 1977. Those simulations will be used to test the behavior of the MAST-1D model under known adjustment configurations and apply prospectively to analyze the morphology and size of the channel response at various artificial recharge scenarios.

6 Conclusion

In this paper we first integrated GIS data within the GAMA platform to validate its potential. We then visualized and interpreted the model outputs from a separate computer model, MAST-1D, to provide a generic visual representation of this model and to spatially display results within the study reach. We introduced a method that uses an agent-based paradigm for visualization purposes through the use of a dedicated language in a specific platform. In this work we described how to separate the simulation from its visualization and showed the benefit of displaying abstract data from a model. With this approach we hope to facilitate and encourage the development of new multi-disciplinary applications by coupling rich visualization with other domains. We are currently working on new ways to interact with a simulation and to share simulations wherein a model could be played at runtime on or in a replay mode at different spatio-temporal scale during the simulation on different devices. The future work faces the challenge of providing new mapping environments for fundamental visual data exploration applied to interaction techniques associated with a variety of scientific domains and a range of complex simulations.

Acknowledgments. The ARCHEM Project (Action de Recherche Collaborative sur les Hydrosystèmes et les Environnements en Mutation) has been supported by the LabEx DRIIHM and the OHM Vallèe du Rhône

References

1. Allan, R.: Survey of Agent Based Modelling and Simulation Tools (2009)
2. Kornhauser, D.U.W., Rand, W.: Design guidelines for agent based model visualization. J. Artif. Soc. Soc. Simul. **12**(2), 1 (2009)

3. Gasmi, N., Grignard, A., Drogoul, A., Gaudou, B., Taillandier, P., Tessier, O., An, V.D.: Reproducing and exploring past events using agent-based geo-historical models. In: Grimaldo, F., Norling, E. (eds.) MABS 2014. LNCS, vol. 9002, pp. 151–163. Springer, Heidelberg (2015)
4. Grignard, A., Drogoul, A., Zucker, J.-D.: Online analysis and visualization of agent based models. In: Murgante, B., Misra, S., Carlini, M., Torre, C.M., Nguyen, H.-Q., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) ICCSA 2013, Part I. LNCS, vol. 7971, pp. 662–672. Springer, Heidelberg (2013)
5. Grignard, A., Taillandier, P., Gaudou, B., Vo, D.A., Huynh, N.Q., Drogoul, A.: GAMA 1.6: advancing the art of complex agent-based modeling and simulation. In: Boella, G., Elkind, E., Savarimuthu, B.T.R., Dignum, F., Purvis, M.K. (eds.) PRIMA 2013. LNCS, vol. 8291, pp. 117–131. Springer, Heidelberg (2013)
6. Lauer, J., Parker, G.: Modeling framework for sediment deposition, storage, and evacuation in the floodplain of a meandering river: Theory. Water Resour. Res. **44**(4) (2008)
7. Lauer, W., Viparelli, E., Piegay, H.: A 1-d size specific numerical model for gravel transport that includes sediment exchange with a floodplain. In: EGU General Assembly Conference Abstracts, vol. 16, p. 10126 (2014)
8. North, M.J., Collier, N.T., Ozik, J., Tatara, E.R., Macal, C.M., Bragen, M., Sydelko, P.: Complex adaptive systems modeling with repast simphony. Complex Adapt. Syst. Model. **1**(1), 3 (2013). http://www.casmodeling.com/content/1/1/3
9. Piégay, H., Radakovitch, O., Fantino, G.: Synthèse des résultats 2010–2013, recommandation opérationnelles et perspectives
10. Railsback, S.F., Lytinen, S.L., Jackson, S.K.: Agent-based simulation platforms: review and development recommendations. Simulation **82**(9), 609–623 (2006)
11. Repenning, A.: Agentsheets: a tool for building domain-oriented visual programming environments. In: Proceedings of the INTERACT 1993 and CHI 1993 Conference on Human Factors in Computing Systems, pp. 142–143. ACM (1993)
12. Resnick, M.: Starlogo: an environment for decentralized modeling and decentralized thinking. In: Conference Companion on Human Factors in Computing Systems, pp. 11–12. ACM (1996)
13. Stefansson, B.: Swarm: an object oriented simulation platform applied to markets and organizations. In: Angeline, P.J., McDonnell, J.R., Reynolds, R.G., Eberhart, R. (eds.) EP 1997. LNCS, vol. 1213, pp. 59–71. Springer, Heidelberg (1997)
14. Tisue, S., Wilensky, U.: Netlogo: a simple environment for modeling complexity. In: International Conference on Complex Systems, pp. 16–21 (2004)

Dynamic Data-Driven Experiments in the Smart Grid Domain with a Multi-agent Platform

Zülküf Genç(✉), Michel Oey, Hendrik van Antwerpen, and Frances Brazier

The Faculty of Technology, Policy and Management,
Delft University of Technology, Delft, The Netherlands
z.genc@tudelft.nl

Abstract. Pervasive information and communication technologies and large-scale complex systems, are strongly influencing today's networked society. Understanding the behaviour and impact of such distributed, often emergent systems on society is of vital importance. This paper proposes a new approach to better understand the complexity of large-scale participatory systems in the context of smart grids. Multi-agent based distributed simulations of realistic multi-actor scenarios incorporating real-time dynamic data and active participation of actors is the means to this purpose. The Symphony experiment platform, developed to study complex emergent behaviours and to facilitate the analysis of the system dynamics and actor interactions, is the enabler.

1 Introduction

In today's networked society many systems are becoming increasingly more challenging to design, in particular large-scale participatory systems, which involve coordination of many autonomous actors distributed in location, time and organisational context. Participatory systems, by nature social technical systems, are used, for example, in crisis management, demand-supply chains, logistics, traffic management, and distributed energy management. A common characteristic of such systems is complexity: indeterminate system behaviour emerging from aggregated activity of many interacting actors. A minor change triggered by local interactions can create cascading effects, propagating throughout a system, giving rise to unforeseen system behaviour. Interrelating causes and effects across a complex system often is beyond human comprehension.

Computer simulations can assist in understanding the dynamics of such large-scale complex systems [5]. However, their potential is limited by two factors: (I) a massive parameter space makes it difficult to accurately model those systems, especially in the presence of real-time dynamic conditions [9] (II) a large number of independent decision makers makes it impossible to capture all interactions with an adequate level of detail. Simulation tools are often used to analyse local behaviours in a specific part of a complex system. Analysis of overall system behaviour that emerges from interactions of autonomous actors is a challenge. This paper addresses that challenge by introducing a new approach based on the use of a multi-agent and distributed experiment platform that incorporates

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 121–131, 2016.
DOI: 10.1007/978-3-319-31447-1_8

active participation of actors with dynamic real-time data and simulation models, extending Symphony[1] [16]. The integration of dynamic data and actions of real actors is the contribution of this approach to overcome the limitations of existing simulation tools in the analysis of complex systems.

This paper presents recent experiences in using the initial implementation of Symphony to design experiments for distributed energy management under real smart grid conditions across Europe[2]. In those experiments, autonomous actors interact with each other and manage or use many distributed or local resources in the environment examined.

The remainder of this paper is organized as follows. Section 2 provides an overview of related efforts in the field of simulation, in particular of smart grid simulation. Section 3 discusses the basic characteristics of the smart grid domain. Section 4 explains the elements of the Symphony experiment platform by briefly describing a few implementation details. Section 5 presents a use case scenario implementation deployed across Europe. Finally, Sect. 6 concludes the paper.

2 Related Work

Computer simulations are widely applied to analyse and understand complex systems including smart grids. Many of these simulations are agent-based or discrete-event based, and run in a closed computation environment with component models of the systems designed and evaluated. The Repast Suite [15,17] is an example of one of such systems, providing a toolkit for generic agent-based simulations. Repast Simphony provides an interactive, java-based simulation platform supporting agents written in multiple languages. A high-performance version of Repast supports simulations on parallel machines or clusters to improve performance in time. However, Repast does not directly support the integration of distributed entities (e.g., hardware resources or third-party service providers).

GridLAB-D [6,7] is an agent-based simulation framework specifically designed for the simulation of the power distribution in the energy domain. Devices are modelled as differential equations. It includes pre-defined modules for power flow and controls, end-user appliance technologies, consumer behaviour, and market models. These can be used to test and evaluate control strategies, even at the level of individual devices. The framework runs on a single machine, and has no direct support to integrate real-time data or to include hardware-in-the-loop.

The Power Trading Agent Competition (PowerTAC) [12] is an economic simulator of the smart grid. It is used in a competition where broker agents compete with one another in an energy market, selling electricity to consumers and trading for electricity on the wholesale market. PowerTAC provides models for consumers (e.g., households and electrical vehicles), the distribution utility,

[1] Symphony is not related to Repast Simphony, see Sect. 2.

[2] This work has been done in the context of EIT ICT Labs projects, see http://www.eitictlabs.eu/.

the wholesale electricity market. The competition focuses on experimenting with broker strategies to maximize profit.

The Smart-grid Common Open Research Emulator (SCORE) [19] is a dedicated emulator that supports integration of models for both the power and the communication network in a smart grid. SCORE supports implementation and evaluation of different control strategies. SCORE is based on the open source communication network emulator CORE [3]. The nodes of the emulator can also run on different physical computers, thereby enabling distributed emulations which helps scalability. However, there is no direct support for incorporating data from actual hardware or active participation from actors in a smart grid.

Another category of simulators, called co-simulations or hybrid simulations, enable simulators to be run in parallel. Challenges for such frameworks include the interaction between different simulators and time synchronization. An example of such a framework is the High Level Architecture (HLA) [8]. HLA's federates include both computer simulations and interfaces to human actors and a runtime infrastructure with services such as federation management, time management, and data distribution.

Simulation Message Bus (SMB) [14] is another approach designed to create a loosely coupled architecture for co-simulation of heterogeneous components that supports message routing between clients. These clients can be both simulations and emulations. Synchronization between clients is performed by proxies, as is data provisioning to the outside world. SMB has the ability to incorporate hardware-in-the loop and to use real-time data. Coordination knowledge is, however, very limited.

Mosaik [18] has been designed to facilitate the specification and execution of smart grid scenarios, by composing different simulation models and providing functionality to analyse the results, coordinated by a discrete event-based simulator based on SimPy [20], a discrete-event simulator framework written in Python. Mosaik provides an API to connect existing simulators together and to support data exchange. To the authors' knowledge it does not support distributed simulations.

3 Smart Grid Domain

The electric power grid is currently going through a global modernization effort to enable more reliable, resilient, sustainable and energy efficient electricity delivery [2]. The envisioned power delivery system, a smart grid, incorporates information and communication technologies (ICT) into the transmission and distribution infrastructure of electricity. A smart grid accommodates bidirectional flow of electricity and information supporting distributed generation, active end-user participation and intelligent network components. This two-way energy and information flow in deregulated markets blurs the traditional boundary between producers and consumers and opens up the grid's potential to innovative products and services [10].

A smart grid is a complex participatory system involving active consumers, central and distributed generators, intelligent components, and competing stakeholders. As consumers become prosumers (both producer and consumer), new behaviour patterns in the generation, distribution and consumption of electricity emerge.

A smart grid is, in fact, a system-of-systems inheriting the complexity of its actors' systems. It involves distributed decision-making and coordination, by a large number of autonomous actors who interact with each other, and a multitude of local or remote resources in the environment. Local interactions result in the emergence of global changes [4]. For example, an unusually sunny and windy day may lead to excessive generation of electricity from renewable sources owned by prosumers, causing negative energy prices in the market. These negative prices may further trigger a chain of actions across multiple domains, such as charging storage units and electric vehicles, and possibly overloaded components in transmission or distribution networks. It would be impossible to explicitly model the overall system behaviour emerging from interactions of such a vast number of entities [13].

4 Symphony Experiment Platform

Symphony [16] has been designed as a generic experiment platform to analyse and study complex system behaviours by focusing on data-driven interactions among autonomous actors. Symphony provides a distributed experimentation environment that can integrate real-time dynamic data, existing simulation models, actions of real actors, and intelligent agents that can mimic the behaviour of real entities. The following elements are the key features for such a platform to capture all types of interactions while testing and validating new concepts and solutions before turning them to real-life practises.

– **Agent-based:** Agents model interacting actors in real-world use cases. Agents are, in fact, loosely coupled entities that interact through message passing [13], making it possible to explicitly analyse the effects of interactions. The loosely coupled nature of agents also enables decentralized experiments by distributing agents to remote locations. Actors in separate physical locations are involved in experiments through those distributed agents.
– **Distributed:** To handle the complexity of large-scale, distributed systems, the experiment platform is distributed as well, making it possible to scale up for experiments with a large number of interacting agents, and to study the impact of physical separation with distributed coordination.
– **Dynamic Data-Driven:** The smart grid and similar large-scale complex systems include a multitude of components with real-time dynamic conditions such as renewable generators. A detailed model of a component can never fully represent its real behaviour. In a simulation environment accommodating a large set of those models, the aggregated divergence from real system behaviour can lead to incorrect analyses and false outcomes. Incorporating

real-time data streams from actual resources into experiments can offer more accurate analyses and predictions with more reliable results [9].
- **Open:** Openness, the ability to integrate existing data, third-party services, tools and models, enhances the context of experiments. Dynamic data from some domains may not always be available. Experiments requiring participation from such domains utilize existing data or simulation models.

Symphony, as discussed above is an agent-based, distributed experiment platform that can incorporate dynamic data from real sources and third-party service providers, tools, and models. A high-level description of its architecture is given in [16]. In short, the platform consists of three layers as shown in Fig. 1. The multi-agent platform, AgentScape [1], is the agent operating system deployed. Reusable components support the development of elements for experiments. The final layer contains the specification of a particular experiment with experiment configurations, protocols, and agent implementations. The agent operating system provides the ability to run agents on different machines in physically dispersed locations, making it possible for them to find each other and exchange messages. It also provides security mechanisms and resilience against partial system crashes.

Symphony offers a standard way to set up and specify experiment behaviour. Protocols define the structure of agent interaction during an experiment. Agents with the same protocol instances talk the same language. The set-up of an experiment specifies the locations of agents, clock settings, and experiment-specific configuration information. Symphony takes care of distributing the experiment configuration and provides an experiment clock to allow non-real-time (e.g. sped up) experiments.

The platform provides client or service agents to integrate real data sources, third-party service providers, or other tools into experiments by providing standard web-service interfaces, for agents to join and leave experiments. A library of ready-made protocols and web-services for data logging, agent organisation, and user interaction is provided. There is also a web-service based implementation that enables agents to communicate with external smart grid applications and devices in a standard-compliant way. This facilitates the integration of existing smart grid components into experiments. The data format of the information exchange depends on the experiment context, which is, in smart grid experiments, based on the International Electrotechnical Commission's standard for the design of electrical substation automation (IEC 61850 [11]).

An experiment is built using the available components or custom-made components developed specifically for an experiment. The experiment designer decides which physical locations are to be involved in an experiment, which experiment agents are located where, and which protocols they are to use. An experiment is defined by a combination of configuration files and Java class implementations. After an experiment is started, external clients from different locations may join and leave the experiment at any time.

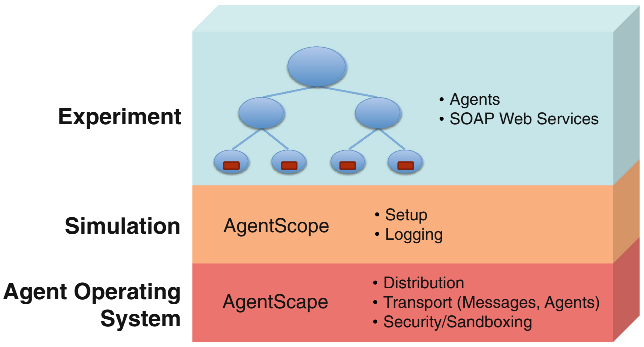


Fig. 1. Symphony platform architecture (taken from [16])

5 A Case Study: Virtual Power Plant Aggregation

A use case study was designed to illustrate the approach proposed in this paper and test the initial version of Symphony. This study involves multiple partners across Europe in the context of Smart Energy Systems Action Line projects of EIT ICT Labs. The use case focuses on the effects of prosumers joining in local energy collectives, forming virtual power plants. Multiple prosumers cluster together into local energy collectives and use storage and demand-response techniques (i.e., shifting energy loads) within the collective to balance energy consumption and production. Different local energy collectives, in turn, negotiate and coordinate their energy consumption and production between themselves.

This experiment scenario combines multiple domains of expertise: market mechanisms, battery storage models, peak optimization, demand side management, energy consumption/production patterns, and distributed negotiations between groups. The purpose of this case study is to demonstrate how modelling, implementation, and analysis of a complex scenario in the smart grid domain are performed with Symphony. Discussion of the actual results of the experiment itself is outside the scope of this paper.

5.1 Scenario Description

Figure 2 illustrates a scenario that has been modelled and implemented. The scenario describes two energy collectives, each consisting of multiple prosumers. Each prosumer, by definition, produces and consumes energy. Some prosumers (e.g., households) have solar panels and others have wind-generators in their neighbourhood. To be able to trade and/or store energy efficiently, prosumers join together in energy collectives.

Each group of prosumers can make a number of local decisions. First, a group can aggregate the combined energy consumption/production of its prosumers. They can optimize the load schedules of all prosumers together to optimize energy costs and/or to avoid use in energy peaks. This is done by shifting

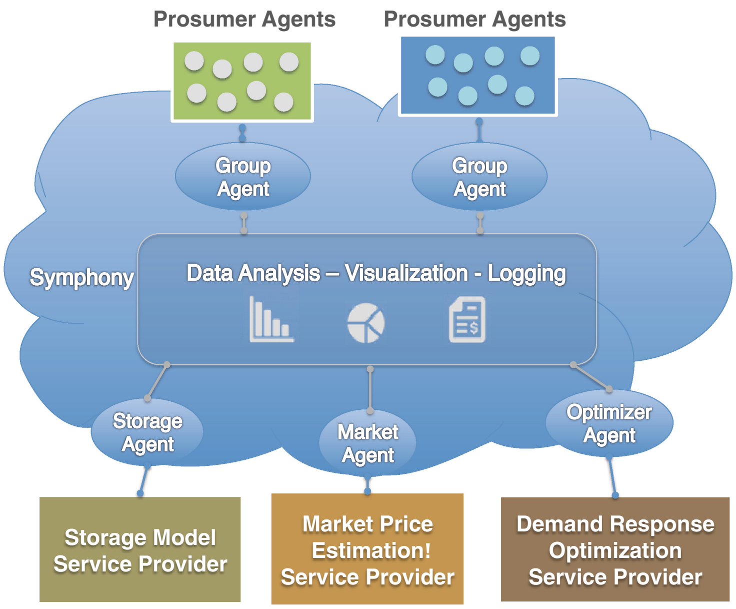

Prosumer Agents
Prosumer Agents
Group Agent
Group Agent
Symphony
Data Analysis – Visualization - Logging
Storage Agent
Market Agent
Optimizer Agent
Storage Model Service Provider
Market Price Estimation! Service Provider
Demand Response Optimization Service Provider

Fig. 2. Distributed scenario: local energy collectives forming virtual power plants.

energy loads to off-peak hours. Shiftable loads are loads generated by appliances, such as washers, dryers, air-conditioners and refrigerators, for which time is not critical. There are constraints when shifting, for example, a washing machine must run before the dryer, and air-conditioners and refrigerators must keep the temperature within a certain range. Last, a group can choose to store any excess generated power in battery storage, so that it can be retrieved at a later time, or it can choose to sell the excess power back to the grid, or perhaps even sell it to other groups that are in need of power.

5.2 Specifying the Scenario in Symphony

The above described scenario is specified within Symphony. Each of the elements in the scenario is represented by agents. The agents also represent resources that interact with sensors or other hardware, such as batteries. A complete list of agents, their behaviour, and interactions in this scenario is as follows:

- *Market Agent* calculates future energy prices given predicted energy loads (based on a 24 h prediction) by using the price estimation service of an external service provider.
- *Storage Agent* provides the storage facility to the prosumers. It uses a storage service from a third-party service provider that has detailed battery models and an actual battery with sensors for hardware-in-the-loop experiments.

- *Optimizer Agent* performs demand-response optimization through an external expert service that calculates an optimal schedule given predicted energy loads, user constraints, and energy price forecasts.
- *Prosumer Agent* represents a prosumer with energy consumption and production (e.g., through PV/wind installations), based on real data.
- *Group (Representative) Agent* represents a group of prosumer agents and interacts with the market, storage and optimizer agents to maximizes the benefits of its prosumers.

Service providers in this scenario are external entities in separate physical locations across Europe. They are connected to the experiment through specific distributed agents. Connections are specified as described in the scenario and summarized in Table 1. The interaction between the group agent and the other agents is not included in this paper for the purpose of simplicity.

Table 1. The agents, their behaviour, and their interactions for the local energy collective scenario.

Agent	Behaviour	Interaction	
		Input	Output
Market	Calculate prices	Load prediction	Price forecast
Storage	Store energy	Storage requests	Battery status
Optimizer	Optimize loads	Shiftable loads	Optimized loads
Prosumers	Predict load	Price forecast	Load prediction
Group	Maximize benefit	...	...

5.3 Implementing the Scenario Within Symphony

The agents and their interactions have been implemented within Symphony. Agents use web services to interact with external entities such as service providers. Agent behaviours have been implemented as protocols and interactions between agents are message-based. The steps performed by the agents in this scenario are as follows:

1. The prosumer agents in a group send their predicted loads with the granularity of 30 min, for the next 24 h, to their group (representative) agent. These load predictions include information on shiftable loads that may be rescheduled (e.g. a washing machine).
2. Each group representative agent sends the combined load/generation predictions to the demand-response optimizer agent to optimize the schedules.
3. The demand-response optimizer agent calculates the optimized load schedules using the price predictions from the market agent for each prosumer agent and sends the result back to the group representative agent.

4. Each group representative agent consults the battery storage agent on the availability of battery storage, either to store excess power or to buy required power. The battery storage agent determines battery usage and price per kWh storage for the group representative agent from the service provider for the storage model.
5. The group representative agents make decisions about storing excess energy or buying energy.
6. The prosumer agents receive their optimized load prediction through the group representative agent.
7. The optimized load is sent to the market agent to determine a new energy price forecast to be used in the next round.
8. The energy prices are sent to the prosumer agents and the cycle starts again.

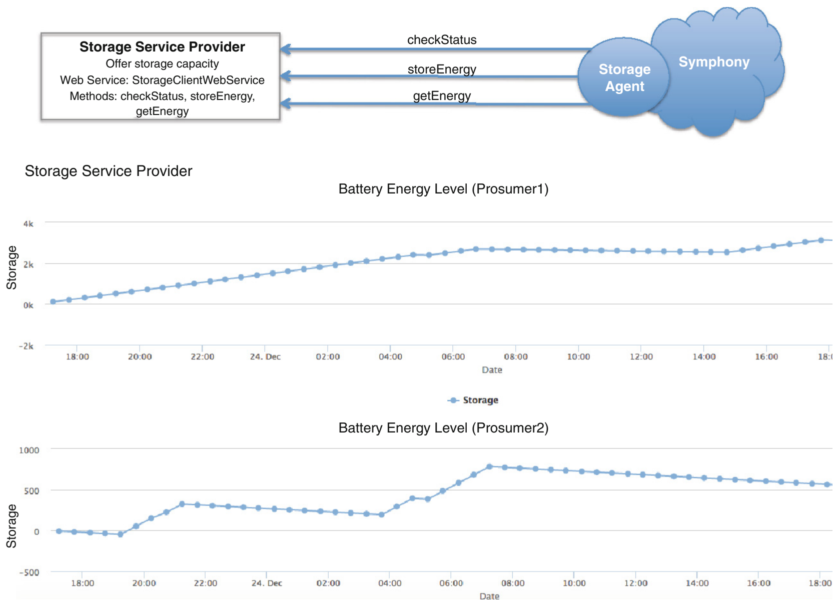


Fig. 3. Interaction of the battery storage agent with the storage service provider and the energy stored for each prosumer group.

Figure 3 depicts the interaction between a storage service provider and a storage agent. It also depicts the battery energy level measured in one of the experiments performed for two of the prosumer group representatives. It shows how different groups deploy batteries in different ways depending on the behaviour of their group members.

In this scenario, the logic is stored within the group agents. In a more elaborate experiment, more groups can be used and the group agents can communicate among each other to share energy. In addition, groups can be dynamic and prosumer agents can join and leave groups during the experiment. This experiment does not take into account the actual transport of energy. It just focuses on the interactions and emergence from algorithms used within the group agents to form and manage a group of prosumers. It is an initial scenario and will be extended with more steps to increase the complexity in the future.

6 Conclusion

This paper presents a new approach to address the complexity in large-scale complex and participatory systems, in particular in the context of distributed energy management for smart grids. The approach is based on modelling and specifying the behaviour of individual actors and their interactions, implementing these specifications for each actor as agents, and analysing their behaviour once implemented in a simulation platform. This platform, Symphony, is agent-based, distributed, dynamic data driven and open, designed specifically to capture emergent behaviours in large-scale complex distributed systems. A use case experiment has been modelled, specified and implemented in the scope of an EIT ICT Labs project involving participants across Europe.

Future research will extend the experimental set-up to develop more use cases with which to analyse different aspects of the smart grid and study emergent behaviours. The approach will also be applied to other domains.

Acknowledgements. Symphony has been partly developed within the EIT ICT Labs projects Open SES Experience Labs (11831) and European Virtual Smart Grid Lab (11814). Partners in these projects include TU-Delft, CWI The Netherlands, Imperial College London, Fortiss, Fraunhofer FOKUS, DFKI, TU-Berlin, and DAI-Labor, to whom the authors are very grateful.

References

1. AgentScape - Distributed Agent Middleware. http://www.agentscape.org/
2. Nist framework and roadmap for smart grid interoperability standards, release 3.0. Technical report, National Institute of Standards and Technology (2014)
3. Ahrenholz, J., Danilov, C., Henderson, T., Kim, J.: CORE: a real-time network emulator. In: Military Communications Conference, MILCOM 2008, pp. 1–7. IEEE, November 2008
4. Amin, S.M., Wollenberg, B.F.: Toward a smart grid: power delivery for the 21st century. Power and Energy Magazine **3**(5), 34–41 (2005)
5. Bar-Yam, Y.: Dynamics of complex systems, vol. 213. Addison-Wesley, Reading (1997)
6. Chassin, D., Schneider, K., Gerkensmeyer, C.: GridLAB-D: an open-source power systems modeling and simulation environment. In: Transmission and Distribution Conference and Exposition, 2008, T&D, pp. 1–5. IEEE/PES, April 2008
7. Chassin, D.P., Fuller, J.C., Djilali, N.: GridLAB-D: an agent-based simulation framework for smart grids. J. Appl. Math. **2014**(12) (2014)

8. Dahmann, J.S., Fujimoto, R.M., Weatherly, R.M.: The department of defense high level architecture. In: Proceedings of the 29th Conference on Winter Simulation, WSC 1997, pp. 142–149. IEEE Computer Society, Washington, D.C. (1997)
9. Darema, F.: Dynamic data driven applications systems: a new paradigm for application simulations and measurements. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004. LNCS, vol. 3038, pp. 662–669. Springer, Heidelberg (2004)
10. Gellings. C.: Estimating the costs, benefits of the smart grid: a preliminary estimate of the investment requirements and the resultant benefits of a fully functioning smart grid. Electric Power Research Institute (EPRI), Technical report (1022519) (2011)
11. IEC-TC57-WG10/11/12. Communications Networks and Systems in Substations. Standard IEC 61850, International Electrotechnical Commission, Geneva, CH
12. Ketter, W., Collins, J., Reddy, P.: Power tac: a competitive economic simulation of the smart grid. Energy Economics **39**, 262–270 (2013)
13. McArthur, S.D., Davidson, E.M., Catterson, V.M., Dimeas, A.L., Hatziargyriou, N.D., Ponci, F., Funabashi, T.: Multi-agent systems for power engineering applications–part i: concepts, approaches, and technical challenges. IEEE Transactions on Power Systems **22**(4), 1743–1752 (2007)
14. Mosshammer, R., Kupzog, F., Faschang, M., Stifter, M.: Loose coupling architecture for co-simulation of heterogeneous components. In: 39th Annual Conference of the IEEE Industrial Electronics Society, IECON 2013, pp. 7570–7575, November 2013
15. North, M., Collier, N., Ozik, J., Tatara, E., Macal, C., Bragen, M., Sydelko, P.: Complex adaptive systems modeling with repast simphony. Complex Adaptive Systems Modeling **1**(1), 1–26 (2013)
16. Oey, M.A., Genc, Z., Ogston, E., Brazier, F.M.T.: Symphony – agent-based platform for distributed smart grid experiments. In: Corchado, J.M., Bajo, J., Kozlak, J., Pawlewski, P., Molina, J.M., Gaudou, B., Julian, V., Unland, R., Lopes, F., Hallenborg, K., García Teodoro, P. (eds.) PAAMS 2014. CCIS, vol. 430, pp. 238–249. Springer, Heidelberg (2014)
17. Repast Development Team. The Repast Suite. http://repast.sourceforge.net
18. Schutte, S., Scherfke, S., Troschel, M.: Mosaik: a framework for modular simulation of active components in smart grids. In: 2011 IEEE First International Workshop on Smart Grid Modeling and Simulation (SGMS), pp. 55–60, October 2011
19. Tan, S., Song, W.-Z., Dong, Q., Tong, L.: SCORE: Smart-grid common open research emulator. In: 2012 IEEE Third International Conference on Smart Grid Communications (SmartGridComm), pp. 282–287, November 2012
20. Team SimPY. SimPy Simulation Package. https://simpy.readthedocs.org

Towards an Agent-Based Model of Passenger Transportation

Banafsheh Hajinasab(✉), Paul Davidsson, Jan A. Persson, and Johan Holmgren

Department of Computer Science, Malmö University, Malmö, Sweden
{banafsheh.hajinasab,paul.davidsson,jan.a.persson, johan.holmgren}@mah.se

Abstract. An agent-based simulation model for supporting the decision making in urban transport planning is presented. The model can be used to investigate how different transport infrastructure investments and policy instruments will affect the travel choices of passengers. We identified four main categories of factors influencing the choice of travel: cost, time, convenience, and social norm. However, travelers value these factors differently depending on their individual characteristics, such as age, income, work flexibility and environmental engagement, as well as on external factors, such as the weather. Moreover, instead of modeling the transport system explicitly, online web services are used to generate travel options. The model can support transport planners by providing estimations of modal share, as well as economical and environmental consequences. As a first step towards validation of the model, we have conducted a simple case study of three scenarios where we analyze the effects of changes to the public transport fares on commuters' travel choices in the Malmö-Lund region in Sweden.

Keywords: Multi-agent based simulation · Traveler behavior modeling · Passenger transport · Impact assessment · Web services

1 Introduction

The design of a "greener" transport system can be supported by a wide set of transport measures, including both transportation policy instruments and investments in infrastructure, such as new public transport pricing schemes, taxes and fares for motorized transport, new bus stops and lines, and new parking space.

In this paper, we propose a novel agent-based simulation model for supporting decision making in urban transport planning. The model, which we refer to as ASIMUT (Agent-based simulator for urban passenger transport), can be used to investigate how different transport measures affect the decisions of the travelers. It takes into account how factors like cost, time, convenience, and social norm influences the decisions on an individual level depending on the socio-economical features of the individual. Another innovative property of the simulator is that it makes use of online web services in order to generate travel options, rather than modeling the transport system explicitly.

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 132–145, 2016.
DOI: 10.1007/978-3-319-31447-1_9

In the next section we review the related work and motivate the chosen agent-based approach. Section 3 presents ASIMUT. To make a first validation of the model, a simple case study of three scenarios is presented in Sect. 4, where we analyze the effects of changes to the public transport fares on commuter's travel choices in a region of Sweden. Some concluding remarks are provided in Sect. 5.

2 Related Work and Motivation

As the application of transport measures may have substantial impact on the travelers' behavior, it is very important to assess their impact before implementation, so that negative effects can be avoided and positive effects can be confirmed. One way of doing this is to perform experimental studies in the real world, but such studies are often very expensive and time-consuming. A common approach for assessing the effects of transport measures is to use computational models, which allows studying the transport system in a simulated environment. A recent review of policy impact assessment models concludes that conventional discrete choice models are the dominating method for travel behavior modeling [8]. These traditional models operate on highly aggregated data. Moreover, they are typically built to study transport in a particular country or a region, and they are often based on the so-called four-step modeling approach. The four steps are: trip generation, where the frequency of trips between zones is determined; trip distribution, where origins are matched with destinations; mode choice, where the proportion of trips between each origin and destination that use a particular transport mode is computed; and route assignment, where all trips are assigned to routes. However, four-step models have been criticized both for neglecting the interaction effects between the involved actors and for oversimplification, which often lead to significant biases in output, especially in settings where the interaction between policies and/or travelers is significant [17]. Furthermore, these models only take into account a limited number of the factors influencing travel behavior [8].

Agent-based simulation modeling is another approach that has been used for impact assessment of transport measures. It is often regarded as a bottom-up approach where each traveler is treated as an interacting, autonomous and independent entity. Thus, it differs from conventional top-down approaches that focus on overall aggregated analysis of the system's behavior [6, 18].

In the agent-based simulation model presented in this paper, the passengers are modeled as agents. We generate the different travel alternatives of an agent using existing web services of online travel planners. We consider both motorized and non-motorized modes of transportation and the combinations of them in generating travel alternatives. The model focuses on how to travel when the destination is already decided, i.e., corresponding to steps 3 and 4 of the traditional four-step models. More specifically, we focus on the mode choice, route choice and departure time choices of travelers, when source and destination data is available from the traveler agent, i.e., the traveler's home and work addresses. We believe that significant improvements to these steps can be made using a more detailed bottom-up approach, and that this can be used together with any approach to determine the travel demand.

The agent-based modeling approach provides a more dynamic approach with respect to the level of detail in modeling different parts. For instance, more interesting parts of the infrastructure can be modeled with a higher granularity. This makes it possible to study the effects of, e.g., building a new bike parking facility that is safe and efficient and close to a train station, or allowing the travelers to bring their bikes on the trains. Furthermore, by using an agent-based method it is possible to model what travel options different travelers actually are aware of, or consider, when deciding what option to choose. This makes it possible to study the effects of, e.g., travel awareness campaigns and the availability of advanced travel planning systems. Such interventions are difficult, or even impossible, to study using traditional models.

Furthermore, agent-based models are able to capture time-related aspects, such as the effects of synchronization and optimization of timetables [16]. There are many transport policy measures that concern time, e.g., time-differentiated congestion and parking fees. Such transport policies are difficult to study using traditional models, but they may have an important influence on travel choices.

We further argue that the use of an agent-based modeling approach, which captures the behaviors of travelers and their interactions between each other and with the environment, will facilitate capturing each individual's preferences and characteristics. This is critically important in order to determine the actual decisions of individual travelers. Thus, agent-based modeling seems very well suited to predict and analyze the effects of different transport measures, since it explicitly models the decisions of each individual and is able to compute the consequences of these decisions. It should be noted that agent-based modeling might require more information about travelers on an individual level than the traditional models, which to a large extent are based on population averages. However, modern consumer technology like smartphones, as well as ITS services like advanced ticketing and tracking systems based on "Internet of Things" technology (connected devices), enable efficient, large-scale, collection of individual travel data.

There are few studies that have applied an agent-based modeling approach in the context of transport policy analysis [8]. In most cases, the agent-based models have been very simple and do not realize the potential of the approach [3, 14]. These models are mostly developed to investigate the effects of a specific transport measure concerning a specific scenario. Furthermore, they do not include all relevant modes of transportation. The input variables, the model construction, and the collected output are very much chosen with a specific scenario in mind. Therefore, these models cannot investigate the effects of various kinds of transport measures in different scenario settings. This means that they are unable to be used as a decision support system to support transport policy making. An agent-based model that bears some resemblance with the one we propose was developed by Grimaldo et al. [7]. It takes into account cost, travel time and environmental in determining travel choice, but it does not regard convenience and makes no difference between individuals (age, income, etc.) except for car-ownership. Moreover, the transport system modeled is very simplistic, e.g., just one road and two travel options, either car or train. In particular, combined transport modes, such as walking, biking, car, bus, and train, are not at all considered.

There are also frameworks for implementing large-scale agent-based transport simulations, e.g. MATSim [4], but they focus on traffic flows and vehicles rather than travel option choices and travelers.

The majority of the traditional models are mode choice models [13], which aim to answer how many travelers will switch to another mode of transport in case of any change in transport system [2]. However, in addition to the choice of transport mode, there are also other important aspects of travel behavior, such as route choice and departure time choice [11]. In order to have a comprehensive and accurate impact assessment, we claim there is a need to investigate the impact on all aspects of travel.

3 ASIMUT

In the proposed model, each passenger is modeled as an agent. This enables us to include each individual's preferences and characteristics into the travel choice modeling. The decision-making process of travelers when choosing between the available travel alternatives is to some extent individual and not the same for all travelers. This means that there is no objectively optimal travel choice from point A to point B for all travelers in a given situation. Therefore, we assume that the "best" travel alternative can be different for different travelers. In ASIMUT, the choices between alternatives are based on four main factors: cost, time, convenience, and social norm. The perceived value (priority) of each of these factors is typically different for each traveler and depends on:

- The traveler's characteristics; refers to the attributes of each traveler and have an important influence on the choice of travel. Examples include socio-economic attributes and geographical location of home and workplace.
- The available travel options at the time of travel and their related cost, travel time, CO_2 emission, number of changes, and walking and cycling distance.
- Contextual factors, factors related to the context where the travel happens, e.g. the current and predicted weather.

Web-services are used in ASIMUT for data collection. We generate the travel alternatives for a traveler from point A to point B, using the web services provided by online travel planners. The use of online travel planners for generating travel alternatives is a novel approach which enables us to capture the most recent information about route alternatives and their relevant characteristics such as cost and travel time. Furthermore, it provides the model with real-time information that adapts automatically with updates, e.g., if the bus schedules change, this change will be automatically updated in ASIMUT. Due to recent developments in application of information systems for online trip planning, nowadays most travelers have access to online travel planners and are able to retrieve almost all the possible travel alternatives at the time of departure. Therefore, we believe integration of web services of online travel planners in ASIMUT makes the model represent the real traveling behavior and is highly consistent with the way travelers choose to travel in everyday life. We use the route alternatives' data gathered from web services as input in the decision-making model.

3.1 Passenger Behavior Modeling

For modeling the individual's travel decision-making we use the theory of planned behavior, which is an extension of the theory of reasoned action [1]. It assumes that humans are rational and they make systematic use of information available to them while they also consider the implications of their actions before they decide for a certain behavior. In ASIMUT, we consider cost, travel time, and convenience as the rational factors that affect the choice of travel. A rational agent aims to maximize the utility and hence minimize cost and travel time and maximize convenience.

However, travelers do not always act completely rational. Social norms and personal values may affect the choice of travel. The theory of planned behavior complements the theory of reasoned action by adding the concept of social norm [1]. Environmental awareness of the travelers is modeled as a social norm in ASIMUT. The theory of planned behavior has also the possibility to cover the behaviors that are not fully under an individual's volitional control. This is very important in travel decision-making where the choice of travel by each individual is not only influenced by her characteristics, attitudes, and subjective norms, but also on intervening environmental conditions, such as the weather which we have included as a contextual factor in ASIMUT.

As mentioned earlier, we use four main categories of factors when making travel choices: cost, time, convenience, and social norms. The significance of each of these factors is determined by each traveler's individual characteristics and contextual factors. In ASIMUT, the value of each of these factors is calculated based on traveler's characteristics and weather conditions. It has been argued that the factors influencing choice of travel can be valued differently for different travel purposes [7, 9]. We have included a weight for each of the factors (i.e., cost, time, convenience, CO_2 emission) in order to be able to change the significance of each factor for different travel purposes. These weights will also be used for calibration purposes. For the decision-making model, we use the weighted sum model [19].

The traveler characteristics that we include in ASIMUT are: age, income, work flexibility, environmental awareness i.e. eco-friendliness, work and home address, working start and end times, access to car, and access to bicycle at home and work. We use work and home address, working hours, access to car, and access to bicycle at home and work directly when generating the travel alternatives, while the other mentioned factors are used for choosing between different travel alternatives. In Table 1 we describe a model of how all these factors can potentially affect the choice of travel and how they interrelate. The main factors influencing travel behavior are listed as columns in Table 1, while the rows are referring to traveler's characteristics and contextual factors. We believe that the income level of the traveler can affect the traveler's perception of travel costs. Therefore, in the proposed decision making model, we use this concept to calculate the value of cost for each traveler; the higher income decreases the influence of the cost on the travel decision of the traveler [5, 13]. For calculating the value of time, we use the traveler's work flexibility factor. We assume that more flexible working hours decreases the value of travel time to some extent.

Johansson et al. show that travelers who are more environmentally conscious tend to take the travel options that have less negative effects on the environments, or more specifically, the travel options that generate least CO_2 emissions [12]. Therefore, in ASIMUT we assume that the amount of CO_2 emission can affect the individual's choice of transport, depending on the individual's level of eco-friendliness.

We assume that convenience is comprised of walking distance, cycling distance, and number of changes for a travel option. The number of changes is defined as the number of transfers between vehicles in order to complete a journey. It has a negative effect on the choice of a travel option; the more interchanges in a travel option, the less convenient it is perceived [10]. Moreover, the number of changes of a travel option makes it less attractive the older you are [15]. Furthermore, we assume that the interchange between vehicles is less convenient in case of bad weather conditions.

Heinen et al. [9] reviewed the factors influencing cycling and indicated that there is a relationship between age and cycling, although it is not universal. While most studies have concluded that the willingness to bike decline with age, there are also some other studies that have not found any significant relation between age and cycling. Weather has also a high influence on the distance the individuals are willing to cycle. High precipitation and low temperature have been found as the most significant weather conditions influencing cycling level. There appears to be no significant relation between the other factors (e.g., income) and cycling [9]. In ASIMUT, we assume that convenience is more important for older travelers. Moreover, bad weather conditions (e.g., rain, snow, or low temperature) decrease the convenience of travel options with long walking distance, cycling distance, and higher number of changes.

Table 1. Interrelationship between the factors influencing choice of travel

Factors		Travel option's attributes					
		Cost	Time	Environ. impact	Convenience		
		Travel costs	Travel time	CO_2 emission	No. of changes	Walking distance	Cycling distance
Traveler's characteristics	Age				*	*	*
	Income	*					
	Work flex.		*				
	Eco-friend			*			
Contextual factor	Weather				*	*	*

3.2 Decision-Making Model

We use a utility function in order to calculate a score for each travel option. The factors influencing travel behavior are the main components of the model. The values of these components are a function of the characteristics of the traveler (i.e., age, income, work flexibility, and eco-friendliness), and contextual factor (i.e. weather). It should be emphasized here that the calculated score actually represents the disutility of a travel option; therefore, an agent will always choose the travel option with the lowest score among the set of available options.

The components of the scoring function have different scales and unit of measurements, and some are quantitative (e.g., age and income), while the others are qualitative or categorical (e.g., weather and work flexibility). In order to avoid domination of larger values, make the components consistent, and neutralize the unit of measurement of the values, we chose to normalize the attributes of the travel options; corresponding to the columns in Table 1. These normalized attributes are referred as relative values in the Eq. (1), e.g., $rel_{oat}^{envImpact}$, which refers to the relative environmental impact of travel option o for agent a. The relative values are typically different for different agents, since these values are calculated with respect to the travel options available for a specific agent. Moreover, we have converted all the characteristics of the travelers and contextual factors to categorical data. These values are called as val_a^{xx} in the Eq. (1), where xx are the factors of the traveler a mentioned in the rows of Table 1 and val_t^{wth} is the value assigned to the weather conditions of trip t. As we discuss further below, all val_a^{xx} and val_t^{wth} are assigned values in the range [0,1].

As mentioned earlier, we chose to assign a weight to each factor, i.e., W_{cost}, W_{time}, W_{conv}, $W_{envImpact}$ refering to the weight of cost, time, convenience, and environmental impact, respectively. These weights are mainly used for calibration, but they can also be used in order to change the importance of each factor according to travel motive, e.g. traveling to work or travel for leisure. The score S_{oat} (i.e., disutility) for travel option o for agent a and trip t is calculated as:

$$\begin{aligned} S_{oat} = {} & W_{cost} * rel_{oat}^{cost} * val_a^{income} + W_{time} * rel_{oat}^{time} * val_a^{workFlex} + W_{conv} * rel_{oat}^{conv} \\ & * val_a^{age} + W_{conv} * rel_{oat}^{conv} * val_t^{wth} + W_{envImpact} * rel_{oat}^{envImpact} * val_a^{eco} \end{aligned} \tag{1}$$

As mentioned earlier, convenience is determined by the three factors of walking distance, cycling distance, and the number of changes of the travel option o for agent a in ASIMUT, and it is calculated as:

$$rel_{oat}^{conv} = rel_{oat}^{wlkDis} + rel_{oat}^{cycDis} + rel_{oat}^{noOfChange} \tag{2}$$

The relative time and cost are calculated by normalizing the cost and time of a travel option with respect to the other travel options of traveler a for trip t. In the below equations, O refers to the collection of all travel options of trip t for traveler a, i.e.,

$$\begin{aligned} & rel_{oat}^{cost} = \frac{Cost_{oat}}{\sum_{o\ O} Cost_{o\,at}}, rel_{oat}^{time} = \frac{Time_{oat}}{\sum_{o\ O} Time_{o\,at} Time}, \\ & rel_{oat}^{envImpact} = \frac{Co2Emission_{oat}}{\sum_{o\ O} Co2Emission_{o\,at}} \end{aligned} \tag{3}$$

The factors for convenience are also normalized, as shown below. For example, in order to calculate the relative environmental impact of a travel option o, the CO_2 emission of that travel option is divided by the sum over the CO_2 emissions of all the travel options o for trip t of the agent a:

$$rel_{oat}^{wlkDis} = \frac{WalkingDistance_{oat}}{\sum_{o\,O} WalkingDistance_{o\,at}}, rel_{oat}^{cycDis} = \frac{CyclingDistance_{oat}}{\sum_{o\,O} CyclingDistance_{o\,at}}, \quad rel_{oat}^{noOfChange} = \frac{NoOfChanges_{oat}}{\sum_{o\,O} NoOfChanges_{o\,at}} \tag{4}$$

As part of the decision-making model, we translate the real values for the age, income, work flexibility, environmental awareness (i.e., eco-friendliness), and weather characteristics, into categories as shown in the Table 2 (in the value column). These translations are the values used in the disutility function, i.e., val_a^{xx} and val_t^{wth}, and they are all numbers between 0 and 1. As an illustrative example, for val_a^{age} we translate an income higher than 100000 SEK to $val_a^{age} = 0.1$, an income in the range [50000, 10000] to the $val_a^{age} = 0.3$, etc. It can be seen that val_a^{age} increases as the income level decreases, which means that the travel cost will be valued lower for the higher income level of the travelers. It should be noted that the values used in the scoring function are just preliminary estimations; they will be further analyzed and validated in future studies.

Table 2. The categorization of characteristics of travelers and contextual factor (val_a^{xx} *or* val_t^{wth})

Variable	Range	Value
Age	15–25	0.1
	25–35	0.3
	35–55	0.5
	55–70	0.7
	+70	0.9
Income (monthly)	+100000	0.1
	50000–100000	0.3
	25000–50000	0.5
	15000–25000	0.7
	<15000	0.9
Work flexibility	high	0.4
	average	0.5
	low	0.6
Eco-friendliness	not concerned	0.3
	medium engagement	0.5
	high engagement	0.7
Weather	Good (no rain or snow, and temp > 10°C)	0.2
	Average (no rain or snow and temp 0-10°C)	0.5
	bad (rain or snow, or temp < 0°C)	0.8

3.3 Generation of Travel Alternatives

For each trip of a traveler, ASIMUT generates a set of travel options, using web services of online travel planners. The attributes that are extracted from the web service, for each travel option include route specification, travel time, cost, CO_2 emission, and the number of changes. These attributes are later used as input data in the traveler decision-making model (see Sect. 3.2 for details).

Waking, cycling, and driving travel options refer to the options that use only one of walking, cycle, and car as the mode of transport all the distance from the origin to the destination. Public transport options refer to the travel options that use public transport together with some short walking to and from public transport stops. They might also include transferring between stops. The time and distance of these short walks are taken into account in the simulation. We further complete the set of travel options by adding additional options where we have replaced long walking distances from origin (A) to a station (A′), and from a station (B′) to destination (B) by cycling. Long walking is defined as walking distances (d) between 200 m and 6000 m. The different travel options from point A to point B are illustrated in Fig. 1.

We use the Google Maps direction API[1] in order to generate walking, cycling, and driving travel options. The cost for the driving option is calculated based on the travel distance and parking fees if the latter apply. To generate the public transport travel options, web services by the public transport providers in the area are needed. In our case, i.e., the most southern part of Sweden, the public transport travel options are provided by the Skånetrafiken Open API[2]. It provides cost, travel time, number of changes, CO_2 emission, and walking distance of each travel option from point A to point B in a specified time and date. We have also used an API called "Commute Greener"[3] in order to calculate the amount of CO_2 emission for car users. The output of the APIs is in XML[4] or JSON[5] schema format. These schemas are parsed in order to extract relevant information, e.g. travel alternatives, travel time, cost, and CO_2 emissions of each alternative.

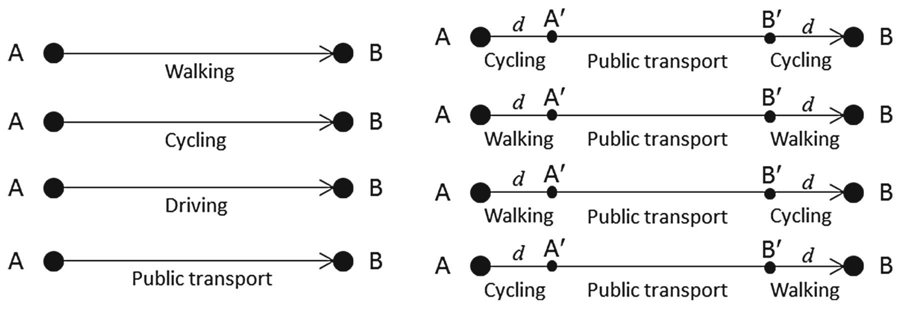


Fig. 1. All considered combinations of transport modes for generating travel options of a trip

[1] https://developers.google.com/maps/documentation/directions/.

[2] http://www.labs.skanetrafiken.se/.

[3] http://developers.commutegreenerinfo.com/.

[4] http://www.w3.org/XML/Schema.

[5] http://json-schema.org/.

When generating the travel alternatives from web services, the characteristics of the traveler are taken into account, i.e., in case the traveler has no access to bike at home, the travel options that include cycling from home will not be generated for that specific traveler, or if the traveler has no access to car, driving options will not be generated. Furthermore, the source and destination of travel options for a specific traveler, and the departure time of the travel are set according to the traveler's information i.e., work/home address and working hours.

Since it is not possible to obtain detailed weather forecast for more than 14 days ahead, we used historical weather data of the same day as the travel date from the last year provided by the Weather Underground service[6]. This service provides temperature, precipitation, and weather conditions (i.e., rainy or snowy) of the same day for the last year. The sequence of steps performed by the model is illustrated in Fig. 2.

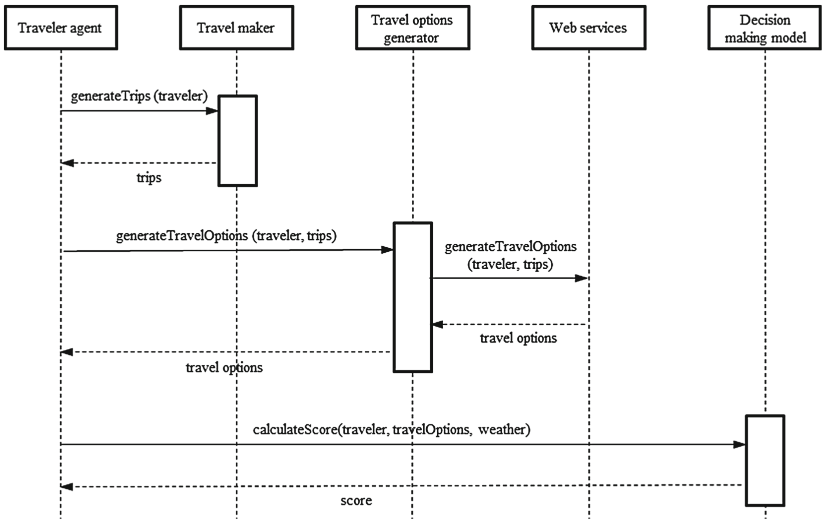


Fig. 2. Sequence diagram of ASIMUT.

4 Case Study

In this section, we present a small case study that is implemented within a prototype of ASIMUT. In this first basic experiment, we use a small sample population of 16 real travelers from the cities of Malmö and Lund in Sweden, who commute between the cities for work and study. This population sample provides the socio-demographic attributes of the travelers, including their work and home addresses.

For each traveler, we generate two trips for commuting to work and back to home respectively, using the traveler's home and work address and work schedule. Travel alternatives are generated for each trip using web services. A score is calculated for each travel option using our decision-making model.

[6] http://www.wunderground.com/.

We study three scenarios; in the first scenario, we simulate the current situation (CS), in the second scenario we examine the effects of reducing the public transport fare to half of the price (HP). The third scenario concerns doubling the public transport fare (DP). We investigate how these changes to the public transport fare are expected to affect the choice of travel and the modal share of the travelers using our implemented prototype. We run the simulation for ten randomly generated days with different weather conditions. The diagrams in Figs. 3 and 4 illustrate how changing the public transport fare is expected to affect the modal share, amount of CO_2 emission (estimated CO_2 footprint per traveler), and travel cost and time for the travelers' commuting during 10 random simulated days. It can be seen from the diagrams that reducing the public transport fare significantly affects the choice of travel and shifts the modal share from private vehicle use to public transport. The walking and cycling share decrease in

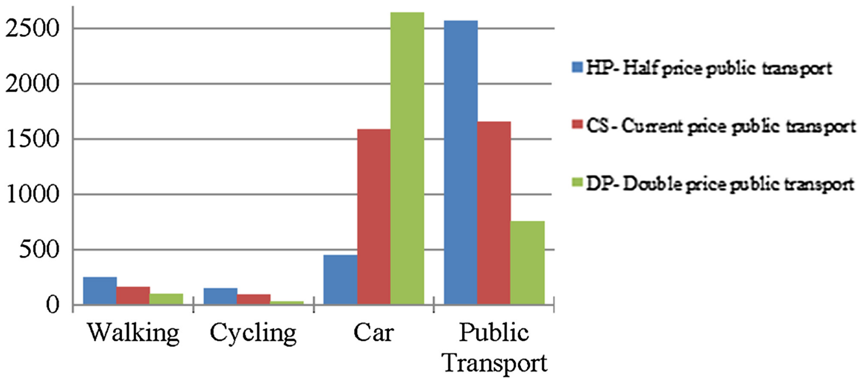


Fig. 3. Modal share (Km) for 10 random days.

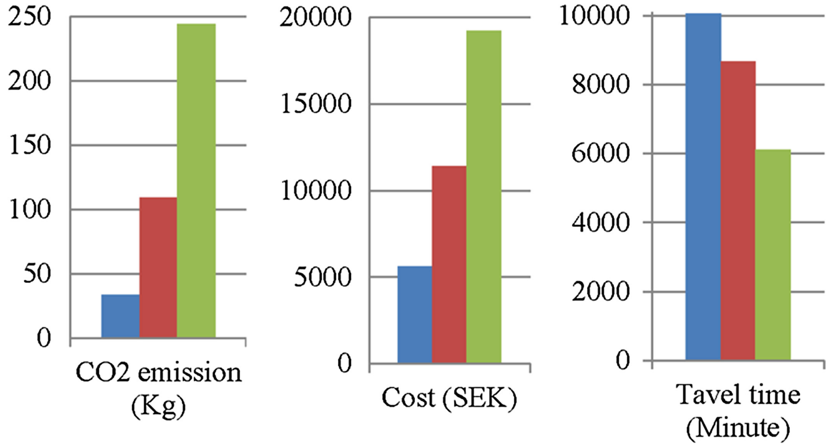


Fig. 4. CO_2 emission, cost and time of selected travel options for 10 random days. Blue = half price public transport, red = current price public transport, and green = double price public transport scenario (Color figure online).

the DP scenario, which we believe is mostly due to the small walking distances between public transport stations, or also due to the travelers who have combined cycling and public transport. When the travelers switch from public transport to private car, the mentioned walking and cycling links will also disappear. Therefore, we observe a decrease in walking and cycling share in the DP scenario. Furthermore, it can be seen from the Fig. 4 that the amount of CO_2 emission is expected to decrease when reducing public transport fare in HP scenario, which can be due to the shift from car use to public transport. Moreover, the selected travel option of the agents cost more when we increase the public transport fare in DP scenario, which can be both because of the increase in public transport fare and the shift to car that is a more expensive mode of transport.

5 Concluding Remarks

This paper has presented an innovative multi-agent based simulation model ASIMUT for modeling travel behavior of passengers. The aim is to support policy makers and urban transport planners in estimating the effects of new transport measures, e.g. policies and infrastructure investments. Some of the characteristics of ASIMUT are:

- It uses combinations of transport modes for generating travel alternatives.
- It uses web services of online travel planners to generate travel options.
- It investigates mode, route, and departure time choice of travelers.
- It considers a range of factors influencing the choice of travel in the travel behavior model, i.e., traveler characteristics, contextual data, and social norm.

Using online travel planners enabled us to access real-time network data that to a large extent corresponds to the data that the real travelers are able to access. It also helped reducing the effort and computation required for generating travel alternatives, calculating travel time, cost, and emissions within ASIMUT. It should also be noted that the use of web services as an input data source may have some potential drawbacks. Firstly, the web services might be temporarily down. Secondly, the performance of web services at a given time might be influenced by the load of the service at that time. Although these potential issues can affect the performance of ASIMUT, we did not notice any of these problems during the development and testing. In order to support the scalability of this approach, we currently cache travel options in order to minimize the number of requests. As a future extension of the approach, we will consider the possibility to run our own server.

We have also described the decision-making model and how the travelers choose between generated travel alternatives. We have included convenience factor in ASIMUT, which is a combination of walking distance, cycling distance and the number of changes in a travel alternative. The initial results from our case study show the feasibility of our approach in travel behavior modeling.

Future work consists of improving the decision-making model in different ways, such as including more factors (e.g. reliability), and investigating the best way to model the correlation between factors, e.g. how income influence the value of travel time. We will also validate the factors considered in the decision-making model and their influence on the travelers' decision-making. The convenience factor can be further developed to include more factors, such as availability of parking facilities. At this stage, social norms only concern environmental awareness, however, this will be further developed in future versions of ASIMUT. We will also investigate the possibility to consider factors like safety and health, for example avoiding walking through parks during night and choosing to walk or bike instead of car or public transport as a choice for healthier life style. The interaction between travelers will also be considered in the further work, e.g. in the form of car-pooling options. We have also planned to apply synthetic population methods in order to generate large populations of realistic agents. Moreover, we will further test ASIMUT through more complicated scenarios, where the effects of combinations of transport measures are investigated.

Future work also includes analyzing the performance of web services, focusing on how the approach scales with increasing number of simulated travelers. In addition, web services typically behave as black boxes, where the users have little (or no) insight in how the services actually operate. To be able to trust the output generated by a model that is based on externally provided web services, it is therefore critical to take special consideration to the output of the web services when validating the model. Future work also includes analyzing issues related to the use of services that cannot directly be validated.

Acknowledgements. We wish to thank K2– The Swedish National Knowledge Centre for Public Transport for partially funding this research, and the National ITS Postgraduate.

References

1. Ajzen, I.: The theory of planned behavior. Organ. Behav. Hum. Decis. Process. **50**(2), 179–211 (1991)
2. Alpizar, F., Carlsson, F.: Policy implications and analysis of the determinants of travel mode choice: an application of choice experiments to metropolitan Costa Rica. Environ. Dev. Econ. **8**(04), 603–619 (2003)
3. An, S., Cui, J.X., Li, L.Y.: Agent-based approach to model commuter behaviour's day-to-day dynamics under pre-trip information. IET Intel. Transport Syst. **5**(1), 70–79 (2011)
4. Balmer, M., Rieser, M., Meister, K., Charypar, D., Lefebvre, N., Nagel, K., Axhausen, K.: MATSim-T: Architecture and simulation times. In: Multi-agent systems for traffic and transportation engineering, pp. 57–78 (2009)
5. De Witte, A., Hollevoet, J., Dobruszkes, F., Hubert, M., Macharis, C.: Linking modal choice to motility: a comprehensive review. Transp. Res. Part A: Policy Pract. **49**, 329–341 (2013)
6. Gilbert, N., Troitzsch, K.: Simulation for the social scientist. Open University Press (2005)
7. Grimaldo, F., Lozano, M., Barber, F., Guerra-Hernández, A.: Towards a model for urban mobility social simulation. Prog. AI **1**(2), 149–156 (2012)
8. Hajinasab, B., Davidsson, P., Persson, J.A.: A survey on the use of computational models for Ex ante analysis of urban transport policy instruments. Procedia Comput. Sci. **32**, 348–355 (2014)
9. Heinen, E., van Wee, B., Maat, K.: Commuting by bicycle: an overview of the literature. Transp. Rev. **30**(1), 59–96 (2010)

10. Hine, J., Scott, J.: Seamless, accessible travel: users' views of the public transport journey and interchange. Transp. Policy **7**(3), 217–226 (2000)
11. Horton, F.E.: Behavioral models in transportation planning. Transp. Eng. J. Am. Soc. Civ. Eng. **98**, 411–419 (1972)
12. Johansson, M.V., Heldt, T., Johansson, P.: The effects of attitudes and personality traits on mode choice. Transp. Res. Part A: Policy Pract. **40**(6), 507–525 (2006)
13. Lave, S.C.A.: A behavioral approach to modal split forecasting. Transp. Res. **3**(4), 463–480 (1969)
14. McDonnell, S., Zellner, M.: Exploring the effectiveness of bus rapid transit a prototype agent-based model of commuting behaviour. Transp. Policy **18**(6), 825–835 (2011)
15. Nitta, Y., Do, G.: Transportation mode-change model to special bus incorporating generalized time. In: 8th International Conference on Transport and Mobility for Elderly and Disabled People (1998)
16. Ramstedt, L.: Transport policy analysis using multi-agent-based simulation, Doctoral Dissertation Series No. 2008:09, Blekinge Institute of Technology, Sweden (2008)
17. Takama, T., Preston, J.: Forecasting the effects of road user charge by stochastic agent-based modelling. Transp. Res. Part A: Policy Pract. **42**, 738–749 (2008)
18. Teodorovic, D.: Transport modeling by multi-agent systems: a swarm intelligence approach. Transp. Planning Technol. **26**(4), 289–312 (2003)
19. Triantaphyllou, E.: Multi-criteria decision making methods a comparative study. Springer, New York (2000)
20. Walle, S.V., Steenberghen, T.: Space and time related determinants of public transport use in trip chains. Transp. Res. Part A: Policy Pract. **40**(2), 151–162 (2006)

Exploring Agent Architectures for Farmer Behavior in Land-Use Change. A Case Study in Coastal Area of the Vietnamese Mekong Delta

Quang Chi Truong[1,2(✉)], Patrick Taillandier[3], Benoit Gaudou[4], Minh Quang Vo[1], Trung Hieu Nguyen[1], and Alexis Drogoul[5]

[1] CTU/IRD JEAI DREAM, CENRES Can Tho University, Can Tho, Vietnam
{tcquang,vqminh,nhtrung}@ctu.edu.vn
[2] PDI-MSC, IRD/UPMC/Sorbonne Universities, Paris, France
[3] UMR IDEES, University of Rouen, Rouen, France
patrick.taillandier@univ-rouen.fr
[4] IRIT, University of Toulouse 1 Capitole, Toulouse, France
benoit.gaudou@ut-capitole.fr
[5] UMI 209 UMMISCO, IRD/UPMC/Sorbonne Universites, Paris, France
alexis.drogoul@ird.fr

Abstract. Farmers are the key actors of land-use change processes. It is thus essential to choose a suitable architecture for farmer behavior to model such processes. In this paper, we compared three models with different architectures to model the farmer behavior in the coastal areas of the Ben Tre province: (i) The first one is a probabilistic model that allows farmer to select the land-use pattern based on land change probability; (ii) The second model is based on multi-criteria decision making and takes into account the land suitability of the parcel and the farmer benefit; (iii) The third model used a BDI (Beliefs - Desires - Intentions) architecture. For each of these models, we have compared the difference between simulated data and real data by using the Fuzzy Kappa coefficient. The results show the suitability of the BDI architecture to build land-use change model and to support decision-making on land-use planning.

Keywords: Agent-based simulation · Agent architecture · BDI architecture · Land-use change · Mekong Delta

1 Introduction

The Mekong Delta region will be heavily influenced by the effects of global climate change [23]. Indeed, the sea level rise and saltwater intrusion will strongly impact the life of people and the situation of agricultural production [9,20]. Nhan *et al.* [12] pointed out that the environmental conditions significantly impact the agriculture and fisheries and that ordinary people tend to spontaneous change the land-use, which causes difficulties for land resource management and cultivation of farmers. Another difficulty comes from the behaviors of farmers that

© Springer International Publishing Switzerland 2016

tend to adapt their food production to the market [19]. The question for planners is how to simulate the behaviors of the farmer to understand land-use and land cover (LULC) change in the next years to build a successful land-use plan. In this context, the choice of an agent architecture to model the farmer behavior is particularly important.

In this paper, we propose to compare three classic agent architectures defined to model human beings behaviors: the first model is based on a probabilistic model, the second one is based on multi-criteria decision making and the last one is based on the BDI paradigm. The comparison is done on an example model simulating the land-use change due to farmers' decisions in the coastal areas of the Ben Tre province (Mekong Delta, Vietnam). The objective is to determine among these three architectures the one that seems the most adapted to model the farmer behavior.

This paper is organized as follows: Sect. 2 presents a state of the art of the existing models of land-use change and an overview of agent architectures used to model such processes. Section 3 is dedicated to the presentation of the three farmer models that have been implemented and of results of their comparison. Finally Sect. 4 proposes a conclusion and offers some perspectives.

2 State of the Art

2.1 Modeling of Land-Use Change

For years, the follow-up study of land-use changes have been mainly based on monitoring the changes in the past using diverse tools related to GIS and Remote Sensing. These tools do not allow to model the dynamic but only to describe the changes. Some researchers have also proposed models based on the combination of GIS, Cellular Automata and Markov chain [11,24] to monitor and to predict the land changing in the future. Although these models give good results for monitoring the past time, they show their limitations to predict the future due to complex behaviors of the social actors in the real world that is not captured by these models. Concerning the modeling of actors involved in LULC changes, Agent-Based Models (ABM) have been heavily used [8,14]. However, most of these models remain simple models based on probabilities. They do not allow to take into account the behavior complexity of the various stakeholders.

In order to represent in a more realistic way the behavior of the different stakeholders, some research works have proposed to use BDI (Belief, Desire and Intention) agent architecture [16] for LULC modeling. In this cognitive agent architecture, agents have beliefs (pieces of information they believe to be true in the world), desires (how they desire the world to become) driving their long-term activities, and intentions of short-term actions to perform to make the world compliant with desires they have chosen and which they commit to achieve. Besides that, agents can change their behaviors and update their beliefs according to what they perceive from the environment. This architecture has been applied to simulate human behavior in many different fields [1]. Concerning land-use planning, Behzadi *et al.* [3] have proposed a BDI architecture applied to city planning. In this model, the planer is the main agent of the model and decides most effective plans to apply according to its beliefs and its goals. Another BDI architecture, based on the belief theory and on a multi-criteria decision-making process, has been proposed in [18] in yearly cropping plan decision-making. This architecture has the advantages to be quite light (it allows to simulate simultaneously thousands of BDI agents) and easy to use by modelers. However, this architecture does not propose any specific formalism for the definition of beliefs and plans and was very specific to its application context.

Only few models have been developed to simulate LULC changes in Vietnam. Among them, the model presented in [4] concerns the planning scenarios in the northern mountains of Vietnam. Another one aims at studying the evolution of cultivation field patterns in the central mountains [7]. However, these models do not rely on real management data from the province departments of natural resources and their case studies are mostly concentrated in the mountains. Almost no study has been carried out about the Mekong Delta, especially the coastal areas affected by saltwater intrusion whereas there are important needs of this type of models.

2.2 Existing Agent Architectures to Model Land-Use Change

Balke and Gilbert have pointed out in an article presenting 14 agent architectures [2] how human beings are very complex machines and difficult to model. However, many agent architectures have attempted to model them. These architectures are often divided in two groups:

- *Reactive architectures.* The agent directly responds to the environment stimuli, without taking into account event history. This architecture is often used in large-scale model. Many of these architectures use production rules where the behaviors are based on "if - then" rules [13].
- *Cognitive architectures.* These agents have reasoning capabilities. They can memorize the past and display complex social behavior. The BDI architecture belongs to this group.

If many complex agent architectures have been proposed, it is not always essential to use them to model the human behaviors, as the human decision-making process is often based more on many sources of information rather than on complex deliberations [5]. Concerning LULC modeling, even if some models propose to use a BDI agent architecture [3,18], most of the are still using simple architectures based on probabilities or on multi-criteria decision-making algorithms [8,14]. The main aims of this paper are thus to compare these three types of architectures on a case study concerning the coastal area of the Vietnamese Mekong Delta and establish recommendations for LULC modeling.

3 Comparison of Agent Architectures to Model Actors Involved in LULC Changes in Coastal Area of the Mekong Delta

3.1 Land-Use Change in Coastal Area of the Vietnamese Mekong Delta

In Vietnam, the land-use zoning policy that defines the type of developments allowed on parcels is defined for 10 years and at four administrative levels: country, province, district and village. The zoning policy is detailed by two plans (five years per plan) under instructions of the Circulars of Ministry of Natural Resources and Environment [10, 22]. However the province land-use changes often do not fit with plans. This phenomenon is particularly visible for the Ben Tre province: the total cultivated area planned for 2010 was 175,824 ha, and reached in reality 179,671 ha (102 %); the rice area reached 38,000 ha compared with the 30,000 ha planned; the aquaculture land was planned to be 39,200 ha but only reached 30,289 ha; at last the forest area only reached 1.30 ha compared with 350 ha planned [15]. This difference of planned and real developments can also be observed at the village level. This unpredictability of the land-use change accentuates the need of reliable tools to simulate the land-use changes to support decision-making process.

3.2 Land Suitability Analysis

As stated in the previous Section, it is important to be able to understand processes underlying land-use changes at the village level in order to be able to predict the evolution of land-use change at province level. In this context, we chose to study the evolution of land-use in the village of Binh Thanh. This coastal village of the Ben Tre province of the Mekong Delta is representative of the region as it contains a mix of brackish and fresh water. In such area the land-use is strongly impacted by the irrigation.

We have collected data concerning the land-use of each parcel of this village in 2005 (Fig. 1) and in 2010 from the Department of Natural Resources and Environment of the Ben Tre province. In this area, six **land-use** types have been defined:

- Rice,
- Rice - Vegetable,
- Rice - Shrimp,
- Annual crops,
- Industrial Perennial tree,
- Aquaculture.

We collected as well the soil map, the saltwater map and the flood map of the regions and define from them six **land-unit** types (a value per land-unit type). From each of these land-unit types, we defined with the help of domain-experts a suitability value for each land-use type. This suitability, which was evaluated by using fourth values, represents the adequacy between the land unit type and the land-use type. For instance, producing industrial perennial tree on a salty soil is very difficult and the yield will be very low.

We also built with domain-experts a transition matrix for each land-use-type. This matrix allows to represent the technical difficulty to change from one land-use type to another one. This difficulty was evaluated using three values (1: easy, 2: medium, 3: very difficult).

At last, we collected data concerning the evolution of benefit and cost of each land-use type for a hectare from 2005 to 2010.

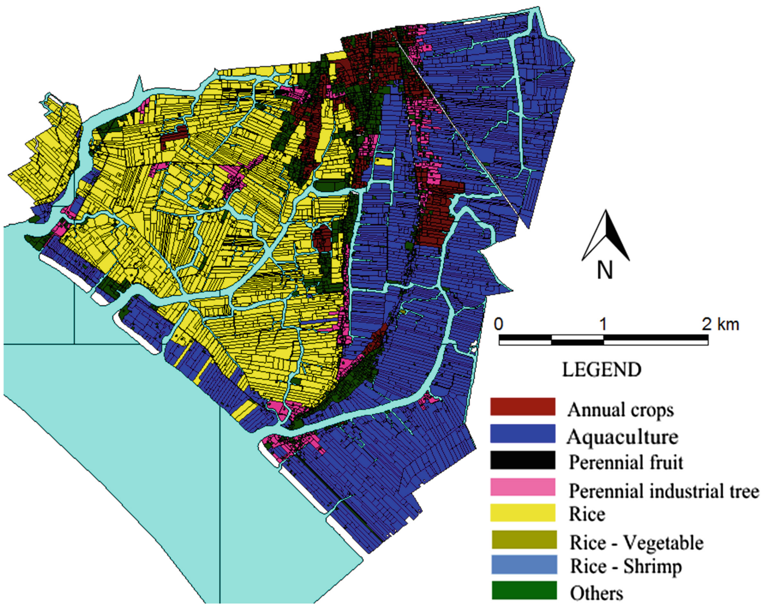


Fig. 1. Land-use map of Binh Thanh in 2005. *source: department of environmental and natural resources of Ben Tre province, Vietnam.*

3.3 Tested Agent Architectures

The next section describes the general model that was used for the test. We describe then the three architectures that have been implemented (the simplest one based on transition probabilities, a more complex architecture) based on multi-criteria decision-making process and the most complex BDI architecture. In this work we only focused on simple architectures with few parameters, as we aim at testing different kinds of architecture and not at defining a very complex and realistic model.

Model Description. The model was defined in order to simulate the evolution of the land-use of the Binh Thanh village. We make the assumption that each farmer has only one parcel and that he has to make a decision concerning the land-use of the parcel every year. A simulation step in this model represents thus 1 year.

In this simple model, the main species of agents is the parcel species that represents the farmer and his/her parcel. A parcel agent has the following attributes:

– *shape*: geometry of the parcel (static),
– *land unit type*: type of soil for the parcel (static),
– *region*: is it inside or outside the dykes (static),
– *neighbors*: list of parcels at a given distance that have the same land unit type (static),
– *land-use*: type of production (dynamic).

In addition to parcel agents, we define a world agent that contains all the global variables:

– *profit_matrix*: for each year (from 2005 to 2010) and for each land-use type the benefit that can be expected from 1 ha of production,
– *cost_matrix*: for each year (from 2005 to 2010) and for each land-use type the cost of 1 ha of production,
– *suitability_by_landuse*: for each land unit type, the suitability to produce a given land-use type,
– *transition_matrix*: difficulty to change from a land-use to another one.

At each simulation step (i.e. every year), each parcel agent is activated in a random order. When activated, a parcel agent (embedding the farmer making decision) chooses its new land-use (that can be the same as the previous one) and changes to it if necessary.

The different models presented in the following sections deal with this choice of a new land-use. Note that all the agent attributes defined here will not be used in all the models. For instance the *neighbors* attribute will only be used by the BDI model.

Decisions Based on Transition Probabilities. In the first farmer behavior model, farmer decisions are based on land change probability. A matrix of probabilities to shift from one land-use type to another was built using the data from 2005 and 2010 and more particularly the number of parcels that changed from 2005 to 2010 (Table 1). As the study area is composed of 2 very different regions - one inside the dykes and the other one outside - we chose to create two matrices corresponding to each region.

The decision-making behavior of the agent is very simple: at each simulation step (i.e. each year), a parcel agent has the probability 0.2 to change its land-use. Each time, it chooses to change its land-use, it draws randomly its new land-use according to the matrices previously defined. From the data we can observe that parcels have changed their land-use on average once during the

Table 1. Land-use changed from 2005 to 2010, counted by number of parcels

		Land-use in 2010 (out-inside dykes)					
		A. Crops	Trees	Rice-Shrimps	Rice-Vege	Rice	Aquaculture
Land-use in 2005 (in-outside dykes)	A. Crops	77-134	64-13	39-0	0-6	0-0	8-1
	Trees	20-8	151-29	3-0	0-31	1-0	2-2
	Rice-Shrimps	0-100	0-0	0-0	0-1016	0-0	0-21
	Rice-Vege	0-0	0-0	1-0	0-0	0-0	0-0
	Rice	0-80	35-67	8-0	17-48	0-0	2-2
	Aquaculture	17-4	173-8	1123-2	1-0	7-0	88-2

5 years of observations. As a consequence, we consider that each parcel has the probability 1/5 of changing land use type at each step. As a consequence, during a simulation some parcels can change their land-use multi times and others not, which is the observed process.

Decision Model Based on Multi-criteria. In the second model, the parcel agent decisions are made according to a multi-criteria analysis. This type of decision-making process in often used for land-use change models (see for example [17]). We defined 3 criteria for the decision: the profit, the cost and the transition difficulty. Indeed, it is generally accepted that farmers tend to choose a production that maximizes their profits, that minimizes the cost - avoidance of risky productions - and that are easy to implement. More precisely, the criterion values are computed as follows for a given transition from *old_lu* to *lu* and a given *soil* type (i.e. land unit type) and a *year* number:

$$Profit(lu, soil, year) = \frac{matrix\_profit(lu, year)}{(max\_profit(year) * matrix\_suitability(soil, lu))} \quad (1)$$

With:

$$max\_profit(year) = max(matrix\_profit(lu, year)) \quad (2)$$

$$Cost(lu, year) = \frac{matrix\_cost(lu, year)}{max\_cost(year)} \quad (3)$$

With:

$$max\_cost(year) = max(matrix\_cost(lu, year)) \quad (4)$$

$$Transition(old\_lu, lu) = \frac{(3 - transition\_matrix(old\_lu, lu))}{2} \quad (5)$$

At each year, every parcel agents computes for each of the six possible land-use types the values of the three criteria. Then, using a weighted mean (see

Eq. 6), it chooses the best land-use type (the one that maximizes the fitness).

$$Fitness(lu, soil, year) = W_{profit}Profit(lu, soil, year) + W_{cost}Cost(lu, year) + W_{transition}Transition(old\_lu, lu) \quad (6)$$

Decision Model Based on a BDI Architecture. In this last model, we propose to use a BDI architecture to model the behavior of the parcel agents. The choice of the land-use type will be based on the three criteria defined in the previous section. However, instead of considering that all the parcel agents have a precise idea of the expected profit according to their land unit type, we consider in this model that the knowledge of the agents is imperfect. It is only when they will have tested a particular land-use type that they will know the real profit that could be obtained from it. We consider as well that a parcel agent can give information to their neighbors concerning their profit. Indeed, in Vietnamese village, the social links between farmers are particularly strong and they often share advices and knowledge.

Figure 2 presents the UML activity diagram of our BDI architecture. First, the agent tests a probability to change or not its current plan. If it keeps the plan, it continues to execute it. Otherwise, it tests a probability to change its goal. If it does not keep its goal, it selects a new goal among the desires with the highest priority and adds it to its intention base. If several desires have the same priority, the goal is randomly chosen among them. Once a goal has been selected (or if the previous goal is kept) a new plan is selected among the ones that can be activated (according to the agent desires and beliefs). The selected plan is the one with the highest priority. Similarly to the goal selection, if several plans have the same priority, the selected plan is chosen randomly among them. At last, the selected plan is executed.

Table 2. Mean results for the percentage absolute deviation (the lowest, the best) and the fuzzy Kappa (the highest, the best) for the three models and for 100 simulations

	Probabilistic model	Multi-criteria model	BDI model
PAD	59.17 %	30.62 %	34 %
Fuzzy kappa	0.46	0.54	0.55

In our model, each parcel agent has the following belief and desire bases:

– *Beliefs*: pieces of knowledge concerning the expected profit for each land-use. Each belief corresponds to a land-use type and a profit associated to it.
– *Desires*: the parcel agents can have two types of desires:
  • define the best possible land-use type (with priority = 2),
  • produce a given land-use type (with priority = 1).
– *Intentions*: the agent can activate two following plans to fulfill its desires:

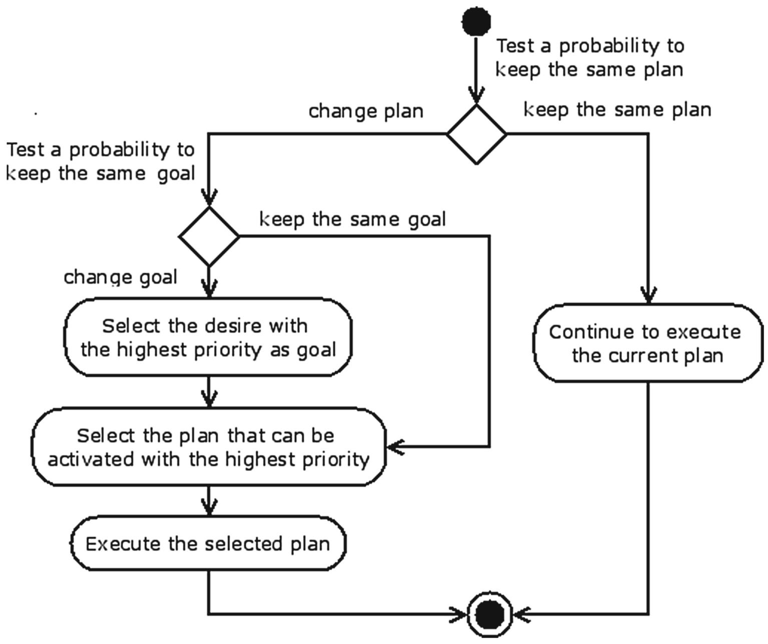

Test a probability to keep the same plan
change plan
keep the same plan
Test a probability to keep the same goal
keep the same goal
change goal
Select the desire with the highest priority as goal
Continue to execute the current plan
Select the plan that can be activated with the highest priority
Execute the selected plan

Fig. 2. Activity diagram of the BDI architecture

- *analyze the possible land-use types*: the agent uses the three criteria and its current beliefs to evaluate each possible land-use type and to define the fittest one. Then the value for the probability to change its current plan is set to 1. The plan is activated when the current goal is *define the best possible land-use*,
- *implement a given land-use type*: the agent sets its land-use attribute to the given one and computes the actual profit with this land-use. Then it updates its beliefs and gives this information to its neighborhood. At last, the value for the probability to change its current plan is set to a given parameter value. The plan is activated when the current goal is *produce a given land-use type.*

3.4 Experiments

The three models were implemented on the GAMA platform[1] [6]. GAMA is an open-source agent-based simulation platform that provides modelers with a complete modeling and simulation development environment to build agent-based simulations. It is particularly powerful concerning the management of GIS data and allows to simply manipulate them.

The different parameter values of the models were defined by using a genetic algorithm to find the parameter set that fits the best to the real data. The fitness

[1] http://gama-platform.org.

function is defined using the Kappa coefficient comparing the observed data and the simulation results in 2010. The fuzzy kappa coefficient allows to compare two maps by taking into account the neighborhood of the parcels [21]. This coefficient is between 0 (not similar at all) to 1 (totally similar). In these experiments, we chose a neighborhood size of 100 m.

Figure 3 shows simulation results obtained from the three models and the observed data. As it can be observed, none of the three models allows to reproduce the exact observed land-use. However, the multi-criteria and the BDI architectures provide better results than the probabilistic one.

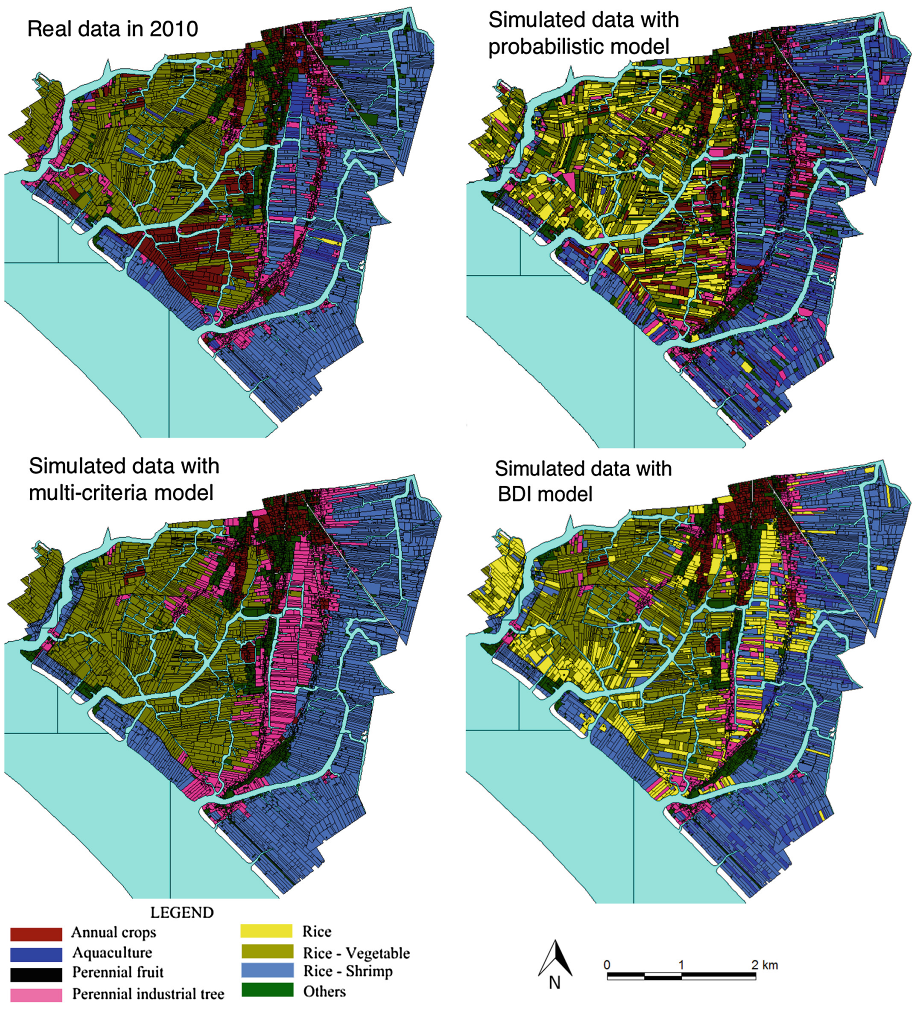

Real data in 2010
Simulated data with probabilistic model
Simulated data with multi-criteria model
Simulated data with BDI model
LEGEND
Annual crops
Aquaculture
Perennial fruit
Perennial industrial tree
Rice
Rice - Vegetable
Rice - Shrimp
Others
N
0
1
2 km

Fig. 3. Observed and simulated land-use map of Binh Thanh in 2010

To quantitatively evaluate the simulation results of the three models, we used two indicators: the fuzzy kappa coefficient (local indicator) and the percent absolute deviation (global indicator).

This second indicator that is often used to evaluate land-use change models is computed by the following formula:

$$PAD(\%) = 100\frac{\sum_{i=1}^{n}|\widehat{X}_i - X_i|}{\sum_{i=1}^{n}\widehat{X}_i} \tag{7}$$

with: $\widehat{X}_i$ the observed quantity of parcels with the land-use i and X_i the simulated quantity of parcels with the land-use i.

As our models (at least two of them) are stochastic, we ran 100 times each model and computed the average fuzzy kappa coefficient (kappa) and percent absolute deviation (PAD). As shown in Table 2, the multi-criteria model allows to get far better results in terms of PAD and kappa indicators than the probabilistic model. Indeed, this model integrates some new pieces of knowledge linked to the economy (profit and cost) and the practices (suitability and transition) that allow to improve the accuracy of the model. Concerning the BDI model, it gives results close to the multi-criteria one (slightly better concerning the kappa coefficient and worst concerning the PAD). However, this model allows to integrate heterogeneity in the model through the influence of the imperfect knowledge and the influence of the neighborhood which make the model more realistic.

4 Conclusion

In this paper, we have compared three architectures to model the land-use change in the Binh Thanh village. The first architecture is based on change probabilities. It gave correct simulation results for reproducing the past events but is limited for predicting the future as it does only allow to reproduce past patterns. The second architecture is based on multi-criteria decision-making. This architecture allowed to get good simulations. However, it does not allow to introduce heterogeneities between agents. The last architecture, which shows the highest fuzzy kappa coefficient, is based on a BDI architecture. This architecture allows to take into account the imperfection of knowledge of farmers and the relations between them. If the three models gave good results, we plan to improve them, and more particularly the third model, by integrating more domain-specific knowledge inside.

References

1. Adam, C., Gaudou, B., Hickmott, S., Scerri, D.: Agents BDI et simulations sociales. unis pour le meilleur et pour le pire. In: RIA, pp. 11–43 (2011)
2. Balke, T., Gilbert, N.: How do agents make decisions? a survey. J. Artif. Soc. Soc. Simul. **17**(4), 3 (2014)

3. Behzadi, S., Alesheikh, A.A.: Introducing a novel model of belief–desire–intention agent for urban land use planning. Eng. Appl. Artif. Intell. **26**(9), 2028–2044 (2013)
4. Castella, J.C., Trung, T.N., Boissau, S.: Participatory simulation of land-use changes in the northern mountains of Vietnam: the combined use of an agent-based model, a role-playing game, and a geographic information system. Ecology Soc. **10**(1), 27 (2005)
5. Dolan, P., Hallsworth, M., Halpern, D., King, D., Metcalfe, R., Vlaev, I.: Influencing behaviour: the mindspace way. J. Econ. Psychol. **33**(1), 264–277 (2003)
6. Grignard, A., Taillandier, P., Gaudou, B., Vo, D.A., Huynh, N.Q., Drogoul, A.: GAMA 1.6: advancing the art of complex agent-based modeling and simulation. In: Boella, G., Elkind, E., Savarimuthu, B.T.R., Dignum, F., Purvis, M.K. (eds.) PRIMA 2013. LNCS, vol. 8291, pp. 117–131. Springer, Heidelberg (2013)
7. Jepsen, M.R., Leisz, S., Rasmussen, K., Jakobsen, J., Muller-Jensen, L., Christiansen, L.: Agent-based modelling of shifting cultivation field patterns, Vietnam. Int. J. Geog. Inform. Sci. **20**(9), 1067–1085 (2006)
8. Mialhe, F., Becu, N., Gunnell, Y.: An agent-based model for analyzing landuse dynamics in response to farmer behaviour and environmental change in the Pampanga Delta, Philippines. Agric. Ecosyst. Environ. **161**, 55–69 (2012)
9. Minh, V.Q., Van, N.T.B.: Simulation of inland water inundation depth under changing of elevation using statistics and spatial interpolation techniques. J. Sci. Cantho Univ. **17a**, 110–117 (2011)
10. MONRE: Circular no. 19/2009/TT-BTNMT: Detailing the establishment, regulation and evaluation planning, land-use planning, Ministry of Natural Resources and Environment of Vietnam (2009)
11. Moreno, N., Wang, F., Marceau, D.J.: Implementation of a dynamic neighborhood in a land-use vector-based cellular automata model. Comput. Environ. Urban Syst. **33**(1), 44–54 (2009)
12. Nhan, D.K., Trung, N.H., Sanh, N.V.: The impact of weather variability on rice and aquaculture production in the mekong delta. In: Stewart, M.A., Coclanis, P.A. (eds.) Environmental Change and Agricultural Sustainability in the Mekong Delta. AGCR, vol. 45, pp. 437–451. Springer, Netherlands (2011)
13. Nilsson, N.J.: A production system for automatic deduction. Technical report, Stanford University (2015)
14. Parker, D.C., Manson, S.M., Janssen, M.A., Hoffman, M.J., Deadman, P.: Multi-agent systems for the simulation of land-use and land- cover change: a review. Ann. Assoc. Am. Geogr. **93**(2), 316–340 (2003)
15. PCBT: Report of land-use zoning in 2020 and the land-use plan for 5 early years 2011–2015 of Ben Tre province, People's Committee of Ben Tre province (2011)
16. Rao, A., Georgeff, M.: Modeling rational agents within a BDI-architecture. In: Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning, pp. 473–484 (1991)
17. Taillandier, P., Therond, O.: Use of the belief theory to formalize agent decision making processes: application to cropping plan decision making. In: European Simulation and Modelling Conference, pp. 138–142 (2011)
18. Taillandier, P., Therond, O., Gaudou, B.: A new BDI agent architecture based on the belief theory. application to the modelling of cropping plan decision-making. In: International Environmental Modelling and Software Society (iEMSs) (2012)
19. Tri, L.Q., Guong, V.T., Vu, P.T., Binh, N.T.S., Kiet, N.H., Chien, V.V.: Evaluating the changes of soil properties and landuse at three coastal districts in Soc Trang province. J. Sci. Cantho Univ. **9**, 59–68 (2008)

20. Tri, V.P.D., Trung, N.H., Thanh, V.Q.: Vulnerability to flood in the Vietnamese Mekong Delta: mapping and uncertainty assessment. J. Environ. Sci. Eng. B **2**, 229–237 (2013)
21. Visser, H., de Nijs, T.: The map comparison kit. Environ. Model. Softw. **21**(3), 346–358 (2006)
22. VNA: Article 24, Land on Law 2003, Vietnamese National Assembly (2003)
23. Wassmann, R., Hien, N.X., Hoanh, C.T., Tuong, T.P.: Sea level rise affecting the Vietnamese Mekong Delta: water elevation in the flood season and implications for rice production. Climatic Change **66**(1–2), 89–107 (2004)
24. Yang, X., Zheng, X.Q., Chen, R.: A land use change model: integrating landscape pattern indexes and Markov-CA. Ecol. Model. **283**, 1–7 (2014)

Optimality and Equilibrium of Exploration Ratio for Multiagent Learning in Nonstationary Environments

Itsuki Noda[1,2(✉)]

[1] National Institute of Advanced Industrial Science and Technology, Tsukuba, Japan
[2] CREST, JST, Saitama, Japan
i.noda@aist.go.jp

Abstract. I investigate relations between total performance of agent societies and relative performance of individual agents with respect to exploration ratio of multiagent learning. The exploration ratio is a key parameter to determine features of multiagent learning in two aspects: as a speed controller of learning in individual agents, and as a reciprocal noise factor for other agents. The investigation figures out trade-off of the two aspects and shows existence of single optimal value of the ratio to minimize the learning errors. I also carried out experiments to compare the performances of agents who use different exploration ratios. The results of the experiments tells existence of equilibrium points to choose the ratio by individual agents. Finally, we discuss the relationship between optimal and equilibrium values of the exploration ratio, which might bring dilemma of selection of the exploration ratio in an evolutionary way.

1 Introduction

Exploration policy is one of key factors to characterize reinforcement learning. Exploration is necessary to get enough experiences for learning agents to adjust their action policies. Therefore, for example, ϵ value in the ϵ-greedy policy should be kept in significantly positive value during the learning. On the other hand, the ϵ should be small in order to avoid loss of rewards for the learning agents. This is called as *dilemma of exploration and exploitation* [4].

In the case of learning under nonstationary environments, the control of the exploration policy become more sensitive. Because agents need to continue learning to adapt to changes of nonstationary environments slowly, they need to continue exploration in a certain amount. Therefore, in the case of the ϵ-greedy policy, ϵ should remain positive rather than conversing to zero.

Multiagent situation brings additional dilemma to select the exploration policy. In the case of multiagent simultaneous learning, exploration of one agent cause noise for learning of other agents. This means that agent should use the small ϵ in ϵ-greedy policies, for example. This direction is inconsistent with ones for nonstationary environments.

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 159–172, 2016.
DOI: 10.1007/978-3-319-31447-1_11

1.1 Related Works and Structure of Article

Because of popularization of mobile information devices and automatic/intelligent appliances, adaptation and balancing of usage of common resources by such equipment becomes an important issue. Traffic control by car navigation systems is a typical example. While each car selects a route from multiple plans, selection by individual cars cause traffic congestion. Scheduling of usage hours of electric powers has also similar problems. These problems are typical application domain of multiagent learning.

Dynamics of multiagent Q-learning are well studied in the case of two-agents and two-actions. Wunder et al. [7] investigated a two-agent game and its adaptation by reinforcement learning with ϵ-greedy exploration. Kaisers and Tuyls [1] applied evolutionary game theory to adapt the stepsize parameter in update rule based on exploration ratio. Main focus of these works is on small-scale (two-agent and two-action) and stationary environments, but not on nonstationary environments.

Tokic and his colleague [5,6] proposed adaption method of exploration ratio using value differences. Their method lets ϵ large when the TD-error is large in a state. They are basically focus only on the case of single-agent and stationary environment.

On the other hand, there are few works on effects of exploration in reinforcement learning under the case of a large population of agents in nonstationary environments. As mentioned above, these situations are general in real-world problems. Especially, reciprocal affects of explorations are serious dilemma.

In this work, we focus on phenomena in learning process under such reciprocal affects in the case of multiagent learning in nonstationary environments. In order to determine the suitable exploration policy for such situation, we need to know the relation between exploration policies and performance of learning of individual/collective agents as the first step. Based on the relation we will be able to discuss about the optimality from the view point of individual and total agent views.

The rest of this article consists of the formalization of repeated nonstationary resource sharing problems (Sect. 2), experimental and theoretical analysis of relations between learning performance and exploration ratio (Sect. 3), and discussions about evaluation of the individual agents using various exploration ratios and the possibility of evolutional methods to find optimal value for exploration ratio (Sect. 4).

2 Repeated Nonstationary Resource Sharing Problem

In this article, I investigate a case of the following *repeated nonstationary resource sharing problems* (RNRSP):

> Multiple agents share some resources. Each agent selects one of resources simultaneously (Fig. 1), while each resource provides an identical reward for all agents who select the resource. The reward is determined according to its capacity value and the number of agents who select the resource.
> The capacities of resources changes slowly on a long-term basis.
> The agents' selection of the resources is infinitely iterated.

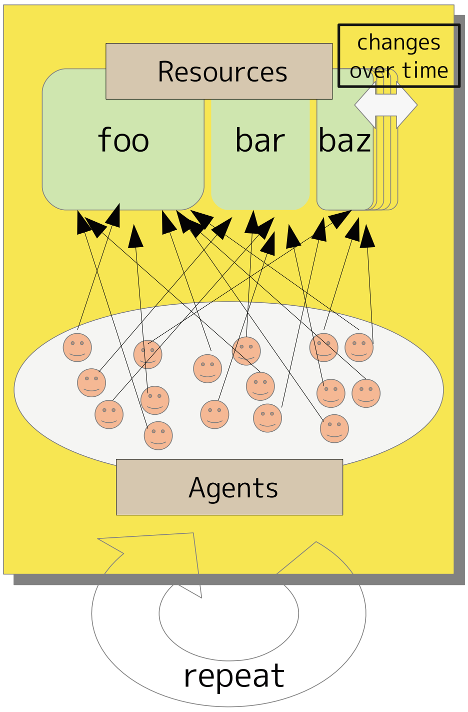


Fig. 1. Repeated nonstationary resource sharing problem

The RNRSP is formally defined as follows:

$$\mathbf{RNRSP} = \langle \mathbf{A}, \mathbf{R}, r \rangle \,, \tag{1}$$

where $\mathbf{A} = \{a_1, a_2, \cdots, a_N\}$ is a set of agents, $\mathbf{R} = \{R_1, R_2, \cdots, R_M\}$ is a set of resources, and $r(R) = f(d_R/\gamma_R) + \text{noise}_t$ is a common reward function. The γ_R and d_R indicate the capacity of each resource R and the population of agents who select the resource R, respectively. We suppose that resources have different capacities, so that the agents should distribute among resources according to the ratio of the capacities. We also suppose that each γ_R changes slowly. This change represents nonstationarity of environments. In the following experiments, γ_R is doubled/halved with a certain small probability (hereafter, referred *fluctuation factors*) in each time step.

The reward function f is supposed to be a monotonically decreasing function. Therefore, when more and more agents select the same resource, the agents get less and less rewards. In the following experiments, I use the function $f(x) = \frac{1}{x}$ (in Exp. 1 and Exp. 2) or $f(x) = \sqrt{\frac{1}{x}}$ (in Exp. 3).

Each agent a has its own reward table $V_a(R)$ that indicates an expected reward for each reward. In every cycle, each agent selects the best resource (in exploitation) or another possible resource (in exploration) on the basis of its own reward table and ϵ-greedy policy. When the agent gets an actual reward from the resource, the it updates its own reward table by a reinforcement learning.

3 Optimal Exploration Ratio to Minimize Learning Error

In this section, I investigate the relation between the exploration ratio and the average learning performance of agents by experiments and theoretical analysis.

In order to measure the total performance of agent learning, I use difference of *agent-distribution* in the following discussions. The *agent-distribution* $\mathbf{d}$ is an array of agent-population d_R. In the case of RNRSP, the ideal distribution $\mathring{\mathbf{d}} = [\mathring{d}_R]$ can be calculated as follows:

$$\mathring{d}_R = \frac{N\gamma_R}{\sum_{R'} \gamma_{R'}},$$

where N is the number of agents. Therefore, the learning performance of the collective agents can be measured by the distance of the ideal and actual distributions. Hereafter, we refer the distance as *learning error.*

3.1 Experimental Analysis

In order to figure out the relationship described above, I conducted an experiment (Exp. 1) of the multiagent learning for a RNRSP.

In the experiment, each agent learns to determine its action policy using reinforcement learning as follow. Each agent a has its own expected reward $V_a(R)$ for each resource R. In the selection of resource, the agent selects a resource using ϵ-greedy policy, where the agent selects resources randomly with the probability ϵ, otherwise, selects the resource whose expected rewards is maximum in the expected reward table V_a. When the agent a selects resource R and gets a reward r, it updates the expected reward for resource R by the following update rule:

$$V_a(R) \leftarrow (1-\alpha)V_a(R) + \alpha r.$$

All agents perform this selection and learning simultaneously.

This experiment used a RNSP that consists of 200 learning agents and 10 resources. I also supposed that all agents share the same learning parameters (the exploration ratio ϵ and the stepsize parameter α), but do not share any other information like expected reward tables. Capacities of the resources are different from each other and slightly changed over time as described above. Therefore the ideal distribution of agents is not stationary.

As defined, the learning performance is measured by learning error, that is, the distance between actual and ideal agent-distributions after 10,000 cycles of agents' selection and learning. Because there are several random factors, I conducted 100 simulations with the same learning and environmental parameters and calculate the average of the learning error. The parameters are set in the following ranges exhaustively: $\epsilon \in (0, 0.5)$ and $\alpha \in [0.0001, 0.3]$.

Figure 2 shows the results of the experiment. In the graph, the horizontal and vertical axis indicate the exploration ratio ϵ and the error (distance between ideal and actual agent-distribution). Each line correspond the case using the same stepsize α. As shown in this graph, the error is large when the exploration ratio is too small. This means that agents can not catch up the changes of environments with too few exploration. On the other hand, large exploration ratio also enlarge the error because of the reciprocal noise factor of exploration. And, the best exploration ratio seems to exist in the middle, where the each error curve hit the bottom.

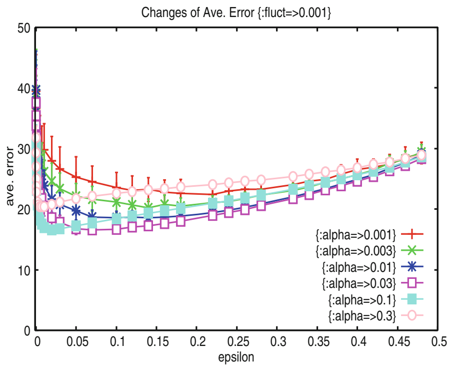


Fig. 2. Changes of distances between ideal and actual agent-distribution for various ϵ (Exp. 1)

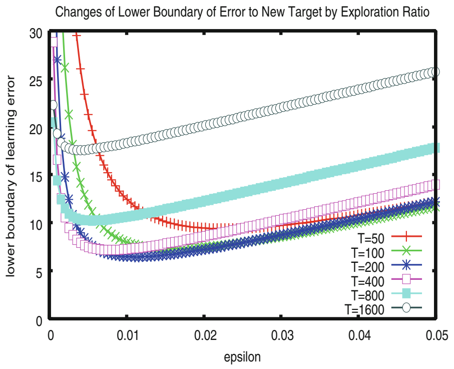


Fig. 3. Lower boundary curves derived by theoretical analysis

3.2 Theoretical Analysis

The characteristics of the relation between exploration ratio and learning error are also supported by a theoretical analysis.

[2] showed a theoretical analysis about relation between lower boundary of learning error and exploration ratio. The analysis finally shows the following corollary about learning error in the MAL situation, which include the case of RNRSP:

Corollary 31. *The lower boundary of learning error of the above MAL situation is given as the following inequality:*

$$Error = E\left[\left\|\mathring{\boldsymbol{d}}'_{\bar{a}} - \tilde{\boldsymbol{d}}'_{\bar{a}}\right\|^2\right] \geq T\sigma^2 + \frac{K\tilde{g}_a}{\epsilon T} + \epsilon N(2 - \frac{K+1}{K}\epsilon), \qquad (2)$$

where $\tilde{g}_a$ is a trail of the inverse Fisher information matrix of advantageous probabilities ρ_a.

Figure 3 shows the relationships between the lower boundary and ϵ. Each curve corresponds to changes in the boundary for different values of T. From the comparison of this graph and Fig. 2, we can find that shapes of both curves are quite similar. Note that the theoretical analysis only shows the characteristics of the lower boundaries but not of the actual error. Because of the similarity, however, we can imply features of effects of exploration to learning error analogically.

It is also known that these tendencies are kept in the case of different populations of agents [3].

3.3 Existence of Optimal Exploration Ratio

Here, I provide an investigation to confirm that there does not exists multiple local minimum but only one optimal ϵ.

Because of the nature of the exploration ratio, it is reasonable to limit the range of ϵ to $[0, \frac{1}{2}]$. If the ratio is greater than $\frac{1}{2}$, the behavior of agents seems almost random. Without the case of the beginning of the learning, such situation is nonsense.

Under this condition, we can prove that the lower boundary of the learning error given by Eq. (2) should be one of the following two cases: (see Sect. A)

– There is a single downward peak of error in the range $[0, \frac{1}{2}]$.
– The error curve is monotonically decreasing.

Therefore, we can focus on the single optimal point and do not care about other local minimum.

4 Heterogeneous Exploration Ratio and Equilibrium

In this section, I explore the case that the exploration ratio is not uniform (hereafter, we call such cases as *heterogeneous* one). In the previous section, the analyses assume that all agents use the same exploration ratio. While this assumption simplifies the analysis, it limits the possibility for individual agents to adjust learning parameters independently. So, in order to investigate the case where agents use different parameters, I conducted the following experiments.

4.1 Gain of Average Benefit

From the viewpoint of individual agents, the main question in the case of the heterogenious exploration ratios is which ratio is superior to other ratio. Consider the case where we apply evolutionary or game-theoretic methods to adjust the exploration ratio. The relative advantage/disadvantage among agents is a key factor to determine the better ratio. If an agent finds that other agents with different exploration ratio are getting better average rewards, it is reasonable for the agent to change the ratio to the other one.

In order to investigate such situation, I conducted experiments with the following setting:

– decide a certain value $\bar{\epsilon}$ as a standard(base) exploration ratio.
– prepare 180 agents who use $\bar{\epsilon}$ as the exploration ratio. (base group)
– prepare 10 agents who use $0.5\bar{\epsilon}$ as the exploration ratio. (small-ϵ group)
– prepare 10 agents who use $1.5\bar{\epsilon}$ as the exploration ratio. (large-ϵ group)

Using the above agents, I carried out an experiment (Exp. 2) using the same condition as Exp. 1.

Figure 4 shows the learning error of Exp. 2. In the experiments, I carried out setting of all combinations of base exploration ratio ($\bar{\epsilon}$) in $[0, 0.5]$, stepsize (α) in $[0.001, 0.3]$, and the fluctuation factor in $[0.0001, 0.03]$. Each line plots the changes of the error to various $\bar{\epsilon}$. Each line corresponds to each values of the fluctuation factor. The lines are grouped by each α value.

In this time, I also evaluated average performances of agents who use different ϵ as follows:

1. calculate the average benefit of agents of each group.
2. calculate ratios of the average benefits of the small/large-ϵ groups to the average benefit of the base group, respectively. (The ratios are refered as *benefit gains.*)

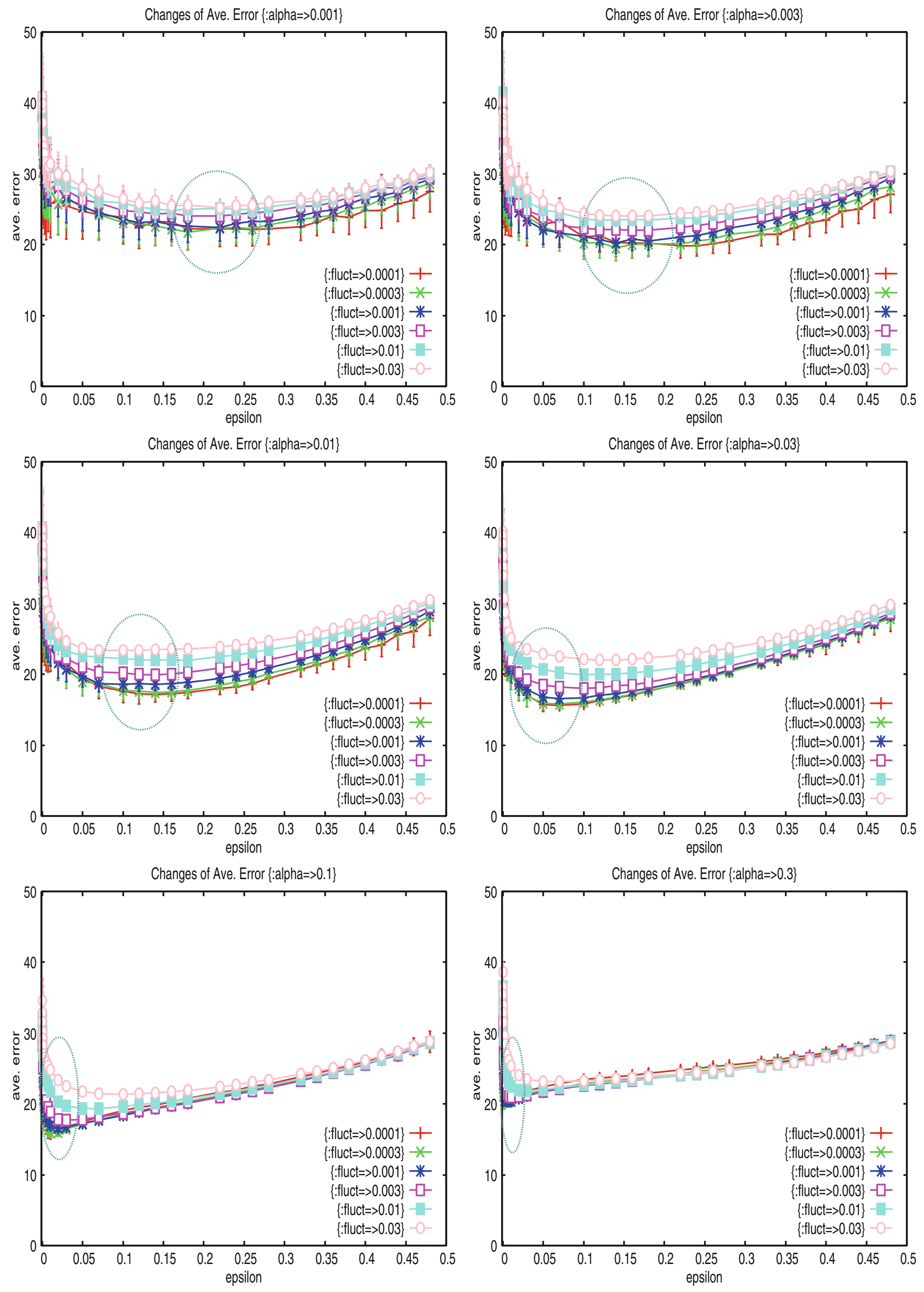
Figure 5 shows changes of the benefit gains to various $\bar{\epsilon}$ values. Each line corresponds to each values of the fluctuation factor. The lines are grouped by each α value. Figure 6 pick-up the lines in the case that the fluctuation factor is 0.0003. In any case, the benefit gain of the small-ϵ group increases when $\bar{\epsilon}$ becomes large, while the gain of the large-ϵ group decreases. Especially, in the cases $\alpha \leq 0.03$, it is clear that the gain of the small-ϵ starts with less than 1.0 for the small $\bar{\epsilon}$, and becomes larger than 1.0 for the large $\bar{\epsilon}$. Symmetrically, the gain of the large-ϵ changes from larger than 1.0 to less than 1.0. Two gains for the same setting cross in the middle.

4.2 Equilibrium by Benefit Gain

Here, we discuss about the possibility to acquire a way to adjust ϵ value using the benefit gain.

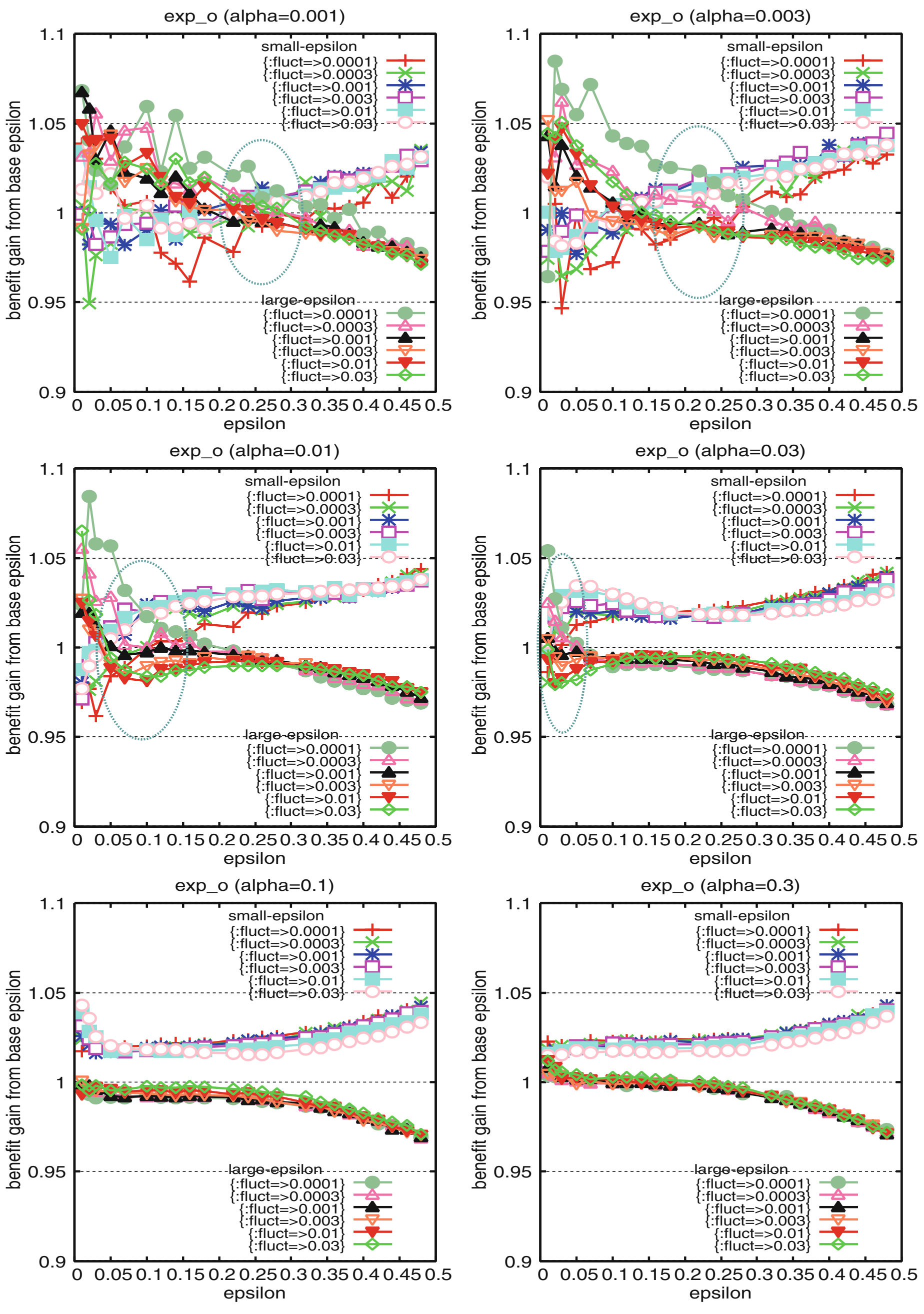
Figure 7 illustrates the changes of benefit gains of small/large-ϵ groups that are acquired in Exp. 2. As discussed in Sect. 4.1, the gain of the small-ϵ group changes from less-than-1.0 to more-than-1.0, while one of the large-ϵ group changes symmetrically. Both lines cross in the middle.

Here, based on this feature, let us try to consider the case to apply evolutionary methods to adjust ϵ to the cross point. Suppose that each agent can know average benefits and ϵ value of other agents. In this case, the agent can imitate ϵ value to the similar value of another agent who gets better benefit. Because of the feature of benefit gain described above, ϵ value of each agent will reach the cross point. When the value is too small, the average benefit of agents in the large-ϵ group is better. So, the agent tends to shift the value to be larger. The symmetric behavior will occur in the case of large ϵ. When the value is too large, the average benefit of agents in the small-*epsilon* group is better.



(The circle in each graphs indicates the optimal area of ϵ.)

Fig. 4. Average error (Exp. 2)



(The circle in each graphs indicates the crossing area of small/large-ϵ groups.)

Fig. 5. Average benefit gain (Exp. 2)

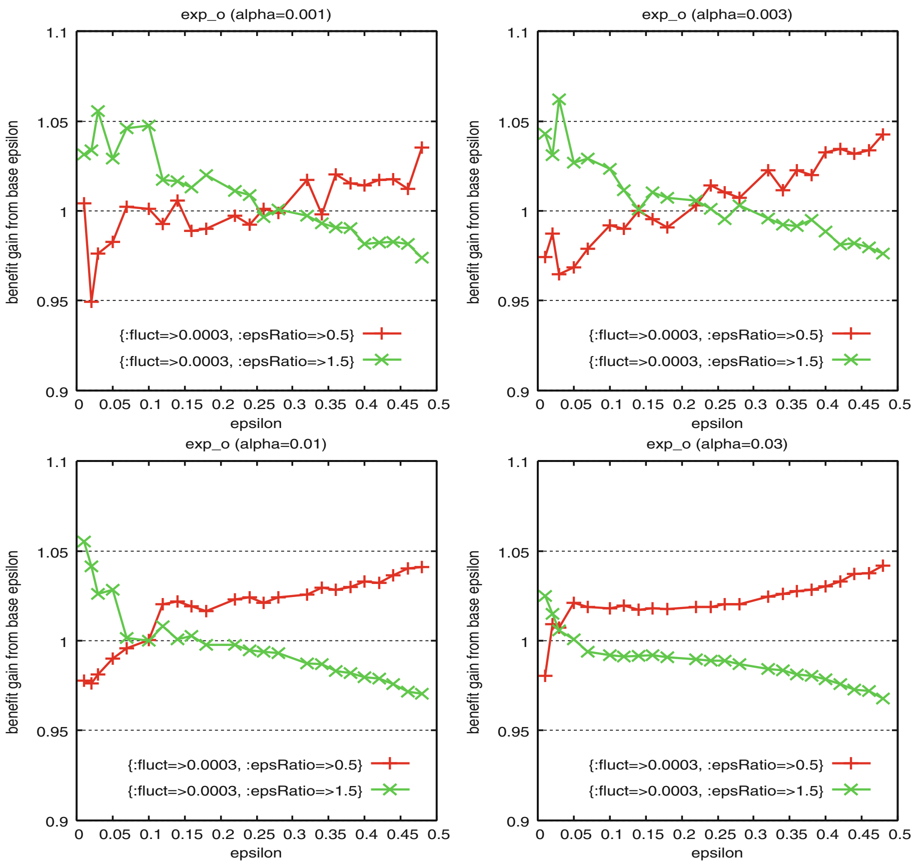


Fig. 6. Average benefit gain (single plot) (Exp. 2)

So, the agent tends to shift the value to be smaller. Finally, the agent will reach the cross point as an equilibrium.

4.3 Equilibrium vs. Optimum

Here, we can have a new question: Whether will the evolutionary adaptation by individual agents described above will reach the optimal value of ϵ for the whole agent society? As described in Sect. 3.3, an agent society with a certain RNRSP has the single optimal value for ϵ. On the other hand, from the discussion in Sect. 4.2, the ϵ value will converge into the equilibrium point when the individual agents changes the value by evolutionary manner. But such equilibrium values are not guaranteed to be optimal from viewpoints of the agent society.

In order to figure out the relation between the equilibrium and optimal value of the ϵ, I analyzed the results of Exp. 2 in Sect. 4.1.

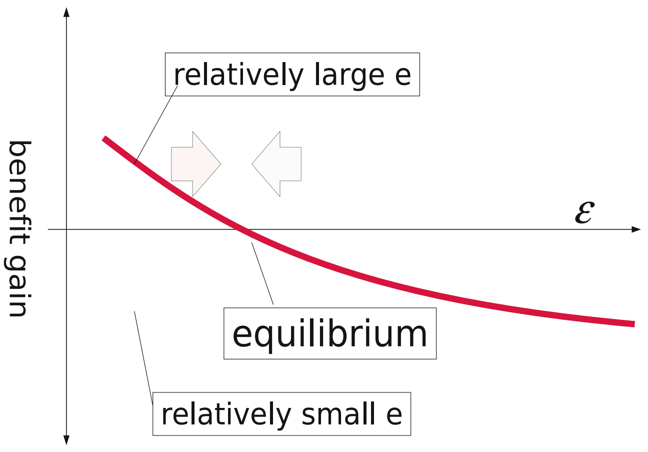
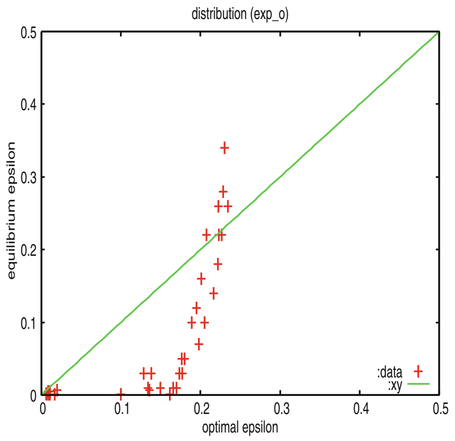


Fig. 7. Illustrated relation of gains using relatively small/large ϵ and equilibrium of ϵ



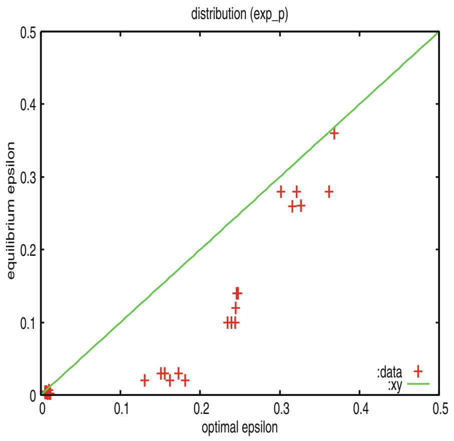


Fig. 8. Relation between optimal ϵ and equilibrium ϵ (Exp. 2)

Fig. 9. Relation between optimal ϵ and equilibrium ϵ (Exp. 3)

Figure 8 shows the relationship between the equilibrium point of ϵ discussed above and the optimal ϵ. Each dot in these graphs corresponds a combination of various fluctuation factors (γ) and stepsize parameters (α). The horizontal axis indicates the value of the optimal ϵ, which minimize the curve of average errors shown in Fig. 4. The vertical axis indicates the value of ϵ at the equilibrium point, where curves of small-ϵ and large-ϵ in Fig. 5 are crossing.

Figure 9 also shows a result of the same analysis of another experiment (Exp. 3), in which we use different type of reward function $f(x) = \sqrt{\frac{1}{x}}$. We can see the similar tendency in both graphs of Figs. 8 and 9.

Relations between optimal and equilibrium is not simple linear: When the optimal ϵ is relatively small, the equilibrium value is smaller than the optimal one and close to zero. When the optimal value becomes relatively large, then, the equilibrium value grows up suddenly. Especially, in the case of Exp. 2 (Fig. 8), the equilibrium value becomes greater than the optimal one.

Such relations implies the following dilemma: Suppose that each agent is selfish and try to change the exploration ratio by imitating other agents who perform better. In a case where situations of environments requires relatively small exploration for agents, agents tend to tune their exploration ratio very small. Because of too-small exploration, agents can not catch up the changes of environments and total and individual performance of agents get worse. On the other hand, when the environment require relatively frequent explorations, the agents tend to explore too much. It also degrades agents' performance.

5 Conclusion

In this article, I investigate relation of exploration ratio in reinforcement learning and learning performance from the viewpoint of optimality and evolutionary aspect. Experimental and theoretical analyses of learning error tell us the simple shape of error curve and existence of single optimal exploration ratio to minimize the learning error. For the evolutionary aspect, experiments using heterogenious agents who have various exploration ratios are carried out. The results of the experiment shows another dilemma of evolutionary methods to find the best exploration ratio in a selfish way.

For the future work, we need to investigate more general formalization of nonstationary environments and variations of agent learning.

Acknowledgments. This work was supported by JST CREST and JSPS KAKENHI 24300064.

A Unimodality of Lower Bound

Here, the lower bound is denoted as $\mathcal{L}(\epsilon)$. Also, $A = \frac{K\tilde{g}_a}{T}$ and $B = \frac{K+1}{K}$ is introduced for the simplicity. So, $\mathcal{L}$ can be defined as:

$$\mathcal{L}(\epsilon) = T\sigma^2 + A\epsilon^{-1} + \epsilon N(2 - B\epsilon)$$

Because of the definition, $1 < B \leq 2$ is hold.

Lemma A1. *$\frac{\partial \mathcal{L}}{\partial \epsilon}$ is unimodal and convex upward.* □

Proof.

$$\frac{\partial^2 \mathcal{L}}{\partial \epsilon^2} = 2A\epsilon^{-3} - 2NB = 0, \qquad \therefore \epsilon = \sqrt[3]{\frac{A}{NB}}$$

Therefore, $\frac{\partial \mathcal{L}}{\partial \epsilon}$ has only one extremal value, and is unimodal.

$$\frac{\partial^3 \mathcal{L}}{\partial \epsilon^3} = -6A\epsilon^{-4} < 0$$

Therefore, $\frac{\partial \mathcal{L}}{\partial \epsilon}$ is convex upward. ■

Lemma A2. *Suppose that $K \geq 3$. If the solution of $\frac{\partial \mathcal{L}}{\partial \epsilon} = 0$ exists in the range $0 < \epsilon < \frac{1}{2}$, the following inequality is hold:*

$$\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}} > 0.$$

□

Proof. Suppose that the solution of $\frac{\partial \mathcal{L}}{\partial \epsilon} = 0$ is ϵ_0:

$$\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\epsilon_0} = -A\epsilon_0^{-2} + 2N - 2NB\epsilon_0 = 0, \qquad \therefore A = 2N\epsilon_0^2(1 - B\epsilon_0).$$

Now, let's consider the value of $\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}}$:

$$\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}} = N(\epsilon - \frac{1}{2})(\epsilon - \epsilon_0^+)(\epsilon - \epsilon_0^-)$$

$$\epsilon_0^{\pm} = \frac{-(B-2) \pm \sqrt{-3(B-2)(B+\frac{2}{3})}}{4B}.$$

Because $K \geq 3$ and $1 < B \leq \frac{4}{3}$ are hold. So, we can get inequalitis, $\epsilon_0^- \leq 0$ and $\epsilon_0 + \geq \frac{1}{2}$. Therefore, solutions of $\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}} = 0$ exist one in range ≤ 0 and two in range $\geq \frac{1}{2}$.

Because $\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}}$ is a cubic function of ϵ_0 and $0 < \epsilon_0 < \frac{1}{2}$, $\left.\frac{\partial \mathcal{L}}{\partial \epsilon}\right|_{\epsilon=\frac{1}{2}} > 0$ is hold. ■

Theorem A3. *$\mathcal{L}(\epsilon)$ is monotonically decreasing or is uni-modal and convex downward in the closed interval $[0, \frac{1}{2}]$.* □

Proof. Suppose that $\frac{\partial \mathcal{L}}{\partial \epsilon} = 0$ has a solution in the closed interval $[0, \frac{1}{2}]$. Using Lemmas A1 and A2, we can say that the solution is only one in the interval.

Also, at the limit $\epsilon \to 0$, $\frac{\partial \mathcal{L}}{\partial \epsilon}$ is negative. Therefore, $\mathcal{L}(\epsilon)$ is uni-modal and convex downward in the closed interval.

If $\frac{\partial \mathcal{L}}{\partial \epsilon} = 0$ has no solution in the interval $[0, \frac{1}{2}]$, $\mathcal{L}(\epsilon)$ is monotonically decreasing because $\frac{\partial \mathcal{L}}{\partial \epsilon}$ is negative. ■

References

1. Kaisers, M., Tuyls, K.: Frequency adjusted multi-agent q-learning. In: Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010), pp. 309–315, May 2010
2. Noda, I.: Limitations of simultaneous multiagent learning in nonstationary environments. In: Proceedings of 2013 IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2013), pp. 309–314. IEEE, November 2013
3. Noda, I.: Robustness of optimality of exploration ratio against agent population in multiagent learning for nonstationary environments. In: Multiagent Interaction Without Prior Coordination (Technical report WS-14-09), pp. 28–34. AAAI, July 2014
4. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
5. Tokic, M.: Adaptive ϵ-greedy exploration in reinforcement learning based on value differences. In: Dillmann, R., Beyerer, J., Hanebeck, U.D., Schultz, T. (eds.) KI 2010. LNCS, vol. 6359, pp. 203–210. Springer, Heidelberg (2010)
6. Tokic, M., Palm, G.: Value-difference based exploration: adaptive control between epsilon-greedy and softmax. In: Bach, J., Edelkamp, S. (eds.) KI 2011. LNCS, vol. 7006, pp. 335–346. Springer, Heidelberg (2011)
7. Wunder, M., Littman, M.L., Babes, M.: Classes of multiagent q-learning dynamicswith epsilon-greedy exploration. In: Frnkranz, J., Joachims, T. (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML 2010), pp. 1167–1174. Omnipress (2010). http://www.icml2010.org/papers/191.pdf

Maximising Influence in Non-blocking Cascades of Interacting Concepts

James Archbold(✉) and Nathan Griffiths

University of Warwick, Coventry, UK
{archbold,nathan}@dcs.warwick.ac.uk

Abstract. In large populations of autonomous individuals, the propagation of ideas, strategies or infections is determined by the composite effect of interactions between individuals. The propagation of concepts in a population is a form of influence spread and can be modelled as a cascade from a set of initial individuals through the population. Understanding influence spread and information cascades has many applications, from informing epidemic control and viral marketing strategies to understanding the emergence of conventions in multi-agent systems. Existing work on influence spread has mainly considered single concepts, or small numbers of blocking (exclusive) concepts. In this paper we focus on non-blocking cascades, and propose a new model for characterising concept interaction in an independent cascade. Furthermore, we propose two heuristics, Concept Aware Single Discount and Expected Infected, for identifying the individuals that will maximise the spread of a particular concept, and show that in the non-blocking multi-concept setting our heuristics out-perform existing methods.

1 Introduction

When autonomous individuals interact, as part of a large population, the propagation of ideas, strategies or infections throughout the population is determined by the composite effect of interactions between individuals. Populations can be viewed as complex systems, with net effects that are hard to predict or influence despite being due to individual behaviour. The propagation of concepts, strategies or infections is a form of influence spread and can be modelled as a cascade from a set of initial individuals through the population.

Understanding how to limit or increase the spread of cascades through a population provides valuable insight into how to influence populations towards a particular state. Such insight has many applications, from informing epidemic control and viral marketing strategies to understanding the emergence of conventions in multi-agent systems. For example, characterising the spread of disease aids in identifying groups of individuals who are at risk, enabling containment efforts to be focused intelligently to avoid wider spread. Understanding how ideas and their adoption propagates can inform viral marketing strategies, or find the network value of individuals in a population. In these cases the key is being able

© Springer International Publishing Switzerland 2016
B. Gaudou and J.S. Sichman (Eds.): MABS 2015, LNAI 9568, pp. 173–187, 2016.
DOI: 10.1007/978-3-319-31447-1_12

to identify the set of individuals who can help to spread an idea or product, or who can restrict future spreading (e.g. through their vaccination).

Several models have been developed to simulate how influence spreads in a network, and much attention has been focused on the *influence maximisation problem*: finding a set of k nodes (individuals) whose activation will maximise the spread of a particular concept. This problem has been shown to be NP-hard, which has led to the development of heuristics to approximate optimal solutions. Many models assume that cascades are *blocking*, in that a node that has been infected/activated by an idea or concept cannot be activated by any others. However, in many domains individuals can hold multiple opinions, adopt multiple strategies, or have multiple interacting infections. The concepts held by an individual will affect those that they are likely to adopt later, and those that they are likely to propagate to others. This informs the idea of cascades or concepts interacting, however most existing work on influence spread has considered single concepts, or small numbers of blocking concepts.

There has been relatively little consideration of cascades with multiple concepts, and such work has made simplifying assumptions. In the domain of epidemic spread Sanz *et al.* developed a model that allows two concepts to interact [20]. The concepts active on a node affect its ability to activate other nodes, and so the spread of a concept is affected by the other concepts within the network. Concept interaction could also be applied in other cascade models, requiring re-evaluation of existing influence maximisation heuristics. There is also the opportunity to develop heuristics that leverage concept interaction to improve concept spread.

In this paper we focus on non-blocking cascades, and propose a new model for characterising concept interaction in an independent cascade. Specifically, we propose a modification to the independent cascade that incorporates interactions for an arbitrary number of concepts. Furthermore, we propose two heuristics, Concept Aware Single Discount and Expected Infected, for identifying individuals that will maximise the spread of a given concept, and we show that in the non-blocking multi-concept setting our heuristics out-perform existing methods.

2 Related Work

In many application areas it would be valuable to leverage influential nodes to maximise the spread of a concept throughout the population. This is referred to as the *influence maximisation problem* where we aim to pick a (minimal) set of nodes that would maximise the spread of information through the population. Several influence propagation models have been proposed in social network analysis literature [8,14]. The target set of nodes is activated at the start of influence propagation, and in subsequent cycles, neighbours of active nodes are activated according some model of propagation. Such models can be classified into two types: those that use node-specific thresholds and those based on interacting particle systems [14].

In the *linear threshold model* [14], a node is influenced by each of its neighbours to varying degrees, as defined by the edge weights. Each node v has a threshold θ_v, such that when the sum of the weights of v's active neighbours exceeds θ_v, v becomes active. Methods have been proposed for maximising influence spread within this model [7], but for now our focus is the independent cascade model.

In the *independent cascade model* (ICM) [10], when a node v becomes active it gets one chance to activate each of its inactive neighbours w, with some probability p_{vw}. Kempe *et al.* showed that a hill-climbing approach can be guaranteed to find a set of target nodes that has a performance slightly better than 63 % of the optimal set [14]. A key issue with the greedy approach is the need to estimate target set quality. Numerous heuristics have been proposed to improve the speed of estimating the influence spread of a node [1,6], but it remains problematic in large networks. Building on the greedy approach, Chen *et al.* proposed a degree discount heuristic that accounts for the existing activations in the network and attempts to reduce the impact of 'double counting' [6]. The degree discount heuristic has been shown to have similar effectiveness to the greedy approaches, while remaining computationally tractable.

The problem of influence maximisation has been studied in many contexts. Early studies into influence spread and maximisation focused on the network worth of users [8,18]. Influence cascades have also been studied in relation to epidemic spread [15,16,20]. The two most commonly applied models when characterising epidemics are the Susceptible Infected Susceptible (SIS) and Susceptible Infected Recovered (SIR) models [5,9]. These both take a probabilistic approach to the independent cascade model, allowing nodes to become deactivated.

Many of these studies have used single cascade models. In many real-world environments, there may be many concepts vying for the attention of an individual. As such, the effect of multiple influence cascades within a single network has been the focus of more recent work on influence spread [11,12], with consideration of competing cascades that model competing products [2], epidemics [13] or general influence spread [3]. Existing work in this area, has typically assumed that the cascades are blocking, meaning that nodes activated/infected by one cascade cannot be activated/infected by another. Additionally, most existing work assumes only two concepts, while in reality there could be many interacting concepts. It is also often assumed that once activated a node remains active, although there are exceptions to this [17].

Sanz *et al.* developed a multi-layer network model in which concepts may only spread on a given layer but nodes can be activated by more than one concept at a time. Other work on the spread of epidemics also limits their travel to a single layer [19]. Existing research typically either assumes blocking concepts on a network layer, or non-blocking concepts that are each limited to a single layer [12]. There has been little consideration of non-blocking concepts in a single layered network. Much of the work in epidemics focuses on the SIS model and the survival thresholds of viruses, with little exploration of multiple concepts interacting within other models [4].

3 Concept Interaction

To model concept interaction, we extend the work of Sanz *et al.* which modelled two interacting diseases [20]. When attempting to infect a susceptible *receiver*, the infectiousness will change if the receiver is infected with the other disease. Conversely, the infectiousness of a disease is affected by the state of the *infector* spreading it. If the infector has both diseases, their infectiousness will change. This model, which was originally intended for use with SIS and SIR cascade models, is the basis for our approach in the independent cascade.

We must allow for both positive and negative relationships when concepts can interact. If a concept c affects c' in a positive way, we call it *boosting*, while if c is *inhibiting* c' then the effect is negative. How concepts spread in a given cascade model will change the exact effect of boosting and inhibiting. In general, boosting a concept makes it more likely to activate on a node and inhibiting makes it less likely. These relationships can be asymmetric: a concept could boost another concept that inhibits it and vice versa.

The relationship between two concepts, c and c' is defined by two *concept interaction factor* (CIF) functions, which describe the effect of the interaction on the infector and receiver respectively. Each concept active on an infector will be able to affect the spread of any other concept active on the infector. Concepts do not act independently in the real world, their combination and interaction will affect which concepts a node spreads, and how infectious that spreading is. We refer to these interactions as the *internal* effect of a given concept on the infector. The function $CIF_{int}(c, c')$ represents how concept c' affects the spread of c when an infector with both concepts active attempts to spread c. For the receiver, we consider the concepts it has active and the *external* concept that attempts activation. The concepts already active on a node will affect how willing it is to adopt new concepts. Concepts a node has already activated may make it more or less amenable to new incoming concepts, affecting the chance of activation success. The function $CIF_{ext}(c, c')$ represents how concept c' affects the chance of a successful attempt by an infector to activate concept c on a receiver with c' active. These functions are both bounded in the range of [-1,1]. If c' inhibits c, these functions return a value below 0, while above 0 indicates a boosting relationship. If c' does not affect c the functions return 0.

Since real-world environments may have more than 2 concepts we must evaluate the effect the infector's *internal* and receiver's *external* environment will have on the concept currently spreading. Two *concept interaction environment* functions characterise these effects, $CIE_{int}(C_n, c)$ and $CIE_{ext}(C_n, c)$ describe the internal and external environment respectively for a spreading concept c and set of concepts active on node n, C_n. These functions will take into account whether each concept in C_n boosts or inhibits c and return a value that represents how the combined effects of all the concepts in C_n affect either the infector's ability to spread c or the receiver's receptiveness to c.

The notion of concept interaction is independent of the cascade model considered. For illustration, in this paper we focus on the independent cascade [14], as it has been the focus for much influence maximisation research, and extend it to account for multiple interacting concepts. In the standard independent cascade, an infector has chance p of making a neighbour active. With multiple concepts this probability is affected by the CIE_{int} function of the infector and the CIE_{ext} function of the receiver. When node n attempts to activate concept c on node m, the probability of success in the interactive independent cascade becomes:

$$p_c^s = p_c * (1 + CIE_{int}(C_n, c) + CIE_{ext}(C_m, c))$$

Where p_c is the baseline probability for that concept. CIE_{int} and CIE_{ext} are bounded to prevent unbalanced boosting compared to inhibiting. Boosting and inhibiting should have similar impact, rather than one offering more significant change. Therefore, we define the *concept interaction environment* functions as:

$$CIE_{int}(C_n, c) = max(-1/2, min(1/2, \sum_{c' \in C_n} CIF_{int}(c, c')))$$

$$CIE_{ext}(C_n, c) = max(-1/2, min(1/2, \sum_{c' \in C_n} CIF_{ext}(c, c')))$$

This means that p_c^s can range between $[0, p_c * 2]$. Since we must consider each node's environment and the resulting effect on the current concepts spread, this value will be calculated for each interaction.

Cascades proceed in rounds, with an initial set of active nodes for each concept. Nodes can be in more than one initial set. Each node in the initial set for a concept will attempt to active that concept on each neighbour that is inactive for that concept. Each successfully activated neighbour will attempt to activate its neighbours in the next round. Nodes make one attempt on each neighbour for each concept they have active, and when no concept activates new nodes the cascade ends. For simplicity in this paper, we adopt the assumption that nodes will never deactivate a concept.

4 Heuristics for Node Selection

Several heuristics have been proposed for influence maximisation, as discussed in Sect. 2. In this section we introduce the main existing heuristics and propose two new methods: Concept Aware Single Discount and Expected Infected, which aim to take advantage of concept interaction.

Degree Based Selection. Degree based selection is the simplest heuristic, and has the advantage of only using attributes of the network, meaning that it is cheap to compute. With the degree heuristic we simply select the k nodes with the highest degree, an approach that has previously been shown to be effective [14].

Single Discount. When a node is added to the selection set, each of its neighbours has a chance to be activated in the first subsequent round of a cascade. However, if it is known that a node will become activated, adding it to the selection set provides no additional value, since that node will be activated regardless of whether it is added to the selection set. This is the motivation behind the single discount heuristic. When a node n is placed into the selection set, the degree of all neighbouring nodes is lowered by 1 to represent their reduced network value (i.e. the number of potential activations they can create has reduced since n is already known to be active). Selection using the single discount heuristic proceeds in rounds, selecting the highest degree node and discounting its neighbours until the desired selection size is reached [6].

Concept Aware Single Discount Heuristic. Introducing concept interaction into the environment requires reconsideration of how concepts spread through a network. Each node can now affect the reach of a concept's spread based on the other concepts they have active. Node degree is typically a good indicator of influence, however in a concept interactive environment this is not always the case. A node with many inhibiting concepts will be less desirable than a node with many boosting concepts if their degrees are equal. Similarly, if a node has many neighbours with active inhibiting concepts, its influence is likely to be low.

We propose a new heuristic, Concept Aware Single Discount (CASD), that weights the degree of a node based on its own concept environment and that of its neighbours, with the aim of providing a more accurate value of node desirability. Specifically, for CASD we define node utility as:

$$U_c(v) = CIE_{int}(C_v, c) + \sum_{n \in N(v)} 1 + CIE_{ext}(C_n, c)$$

where $N(v)$ is v's set of neighbours. Since we are attempting to select nodes that would help to maximise the spread of the targeted concept, the internal environment of a node is a good indicator of node value along with its weighted degree. The external environment of a neighbour of v affects the likelihood of v activating it. The aim of the heuristic is to target nodes with many boosting neighbours and avoid those surrounded by inhibiting nodes. Therefore, $CIE_{ext}(C_n, c)$ is used to increase or decrease the contribution a neighbour makes to the degree of a node. This allows for the concept environment of a node and its neighbours to be considered when evaluating it's worth to the selection set.

Selection proceeds in rounds, with the highest valued node selected each round. When a node n is added to the seed set, neighbour v has its utility updated accordingly:

$$U_c(v) = U_c(v) - (1 + CIE_{ext}(C_n, c))$$

In the same way as Single Discount, we remove the value contributed by the neighbour as it can no longer be activated. Once all neighbours have been updated, the next selection is made, until the required number of nodes is selected.

Degree Discount. Degree discount has been shown to be effective in approaching the optimal solution with reasonable computational overhead [6]. It relies on calculating the expected nodes gained from adding a given node to the selection set. When a node is added, the expected gain of adding its neighbours is lowered. Additionally, those neighbours now have a chance to be activated in the first round of a cascade. The heuristic therefore weights the degree of a node based on these factors, updating the value for any neighbours when a node is added to the selection set. Nodes are initially ranked by degree, and when a node is added to the seed set neighbours have their degree set to $d_v - 2t_v - (d_v - t_v) * t_v * p$, where d_v is the original degree, t_v is the number of neighbours in the seed set and p is the probability of infection. This calculation is based on the expected benefit of such nodes (details of its derivation can be found in [6]).

Expected Infected Heuristic. It is important to consider the environment of a node and its neighbours when selecting nodes. The expected payoff from a node will change if it is surrounded by inhibiting concepts compared to boosting ones. Degree Discount is successful because it considers the expected number of activations for a node to decide its value. However, since it is intended for a single cascade model, it requires updating to consider concept interaction. We propose a new heuristic, Expected Infected, with the aim of accounting for these effects.

For each node, v, we consider the set of neighbours with chosen concept c active, $AN_c(v)$. Each of these neighbours will have a chance to activate v, which if successful removes any additional value v would have. The probability of v having c activated by one of these neighbours, $p_a(c, v)$, is:

$$p_a(c, v) = \sum_{n \in AN_c(v)} OP_c(n, v)$$

The sum of the individual chances of each neighbour to activate concept c on v, known as $OP_c(n, v)$, can be defined as:

$$OP_c(n, v) = p_c * (1 + CIE_{int}(C_n, c) + CIE_{ext}(C_v, c))$$

We can now determine the number of activations from $N(v)$ that can be expected as a result of activating v, as follows:

$$EA_c(v) = 1 + \sum_{n \in N(v) \setminus AN_c(v)} OP_c(v, n)$$

In addition to v itself, for each non-active neighbour, we have a $OP_c(v, n)$ chance to activate concept c. Summing the probabilities for each neighbour gives the expected number of neighbours v will activate. However, the chance that v will be activated by a neighbour must also be considered, and so the expected utility for adding v to the seed set is given by:

$$U_c(v) = (1 - p_a(c, v)) * EA_c(v)$$

where $1 - p_a(c, v)$ is the chance of v not being activated. If activated anyway, v will give no additional value. Accounting for this requires scaling $EA_c(v)$ by the probability that v does not get activated. Initially nodes have a value of $EA_c(v)$, since there will be no active neighbours.

Table 1. Experimental Parameters

Parameter	Values
Graph Type	Small-world, Scale-free
Graph Size (nodes)	1000, 5000
Boost Proportion	0, 0.1, 0.2, 0.3, 0.4
Inhibit Proportion	0, 0.1, 0.2, 0.3, 0.4
Initial set size	1 %, 2.5 % or 5 % of graph size
Intervention set size	1 %, 2.5 %, 5 %, 7.5 % or 10 % of graph size
Rounds before intervention	5, 10, 25

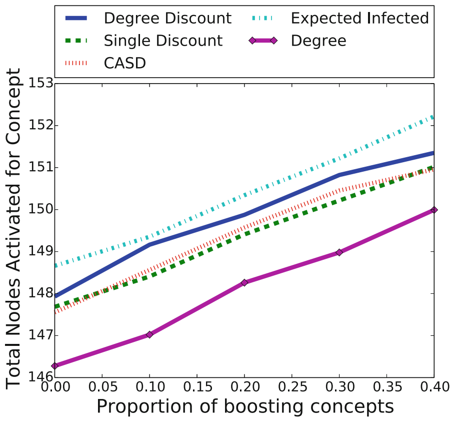
In each selection round, we add the node with the highest value for this heuristic and update its neighbours accordingly, continuing until the selection set is the desired size.

5 Experimental Approach

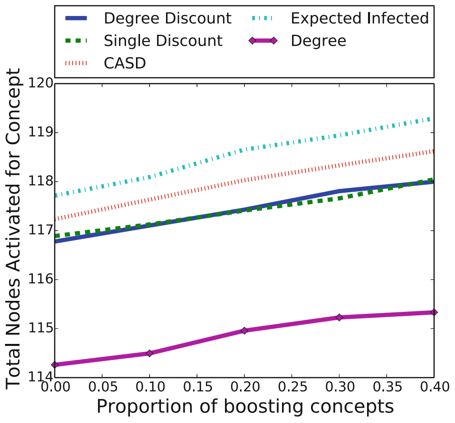
To evaluate the effectiveness of our proposed heuristics in the context of multiple interacting concepts, we perform simulations using the interactive independent cascade model proposed in Sect. 3. Each simulation is performed using 10 concepts, with an activation probability for any concept of 0.05. We use the heuristics introduced in Sect. 4, along with random selection to provide a baseline for comparison. The network topologies listed in Table 1 were used, since they exhibit characteristics found in real-world social networks.

For each simulation, we determine the number of concepts boosted and inhibited by a given concept by selecting from a Gaussian distribution, with a mean of $boost\_proportion * 10$ and $inhibit\_proportion * 10$ respectively, and a standard deviation of 2.5. This, with the proportions defined in Table 1, prevents concepts being too similar and allows for more realistic environments. The final number of concepts boosted or inhibited by a single concept is restricted to be in the range $[0, 5)$.

The initial set of nodes for each concept is selected uniformly at random, and is the size same for all concepts. The cascade proceeds for a fixed number of iterations (a burn-in period) until an intervention occurs, at which point the targeted concept will activate an additional set of nodes selected using a chosen heuristic. The burn-in period before intervention is necessary since the concept aware heuristics require nodes to have concepts activated prior to selection. When selecting intervention nodes, the initial value of a node is discounted considering active neighbours as dictated by the chosen heuristic. This helps to compensate for heuristics that assume no nodes are active at the start. Each heuristic is used for interventions in 100 runs for each combination of parameters in Table 1.



(a) Scale-free network of 1000 nodes



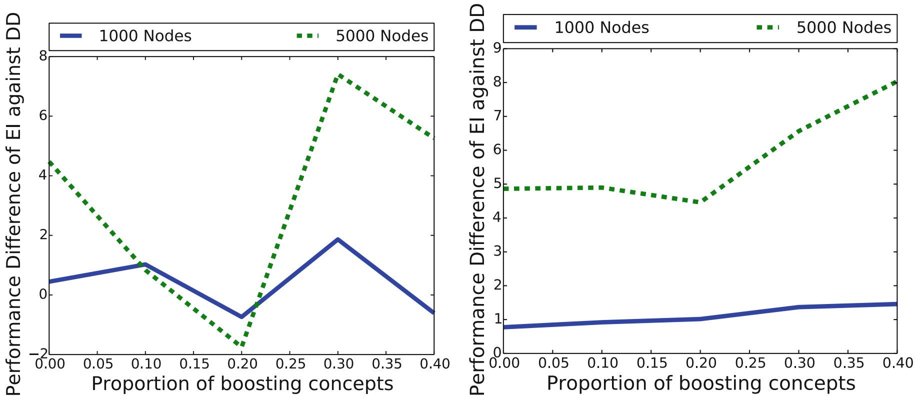
(b) Small-world network of 1000 nodes

Fig. 1. Total activations against proportion of boosting concepts

6 Results

We initially compare the performance of each of the heuristics introduced above for a range of parameters. Random selection performed significantly worse than other heuristics in all cases, and while Degree was less effective than the other heuristics it was by a much smaller margin. Therefore, for simplicity of presentation, we do not consider Random selection further. Figure 1a shows the performance of the heuristics as the proportion of boosting concepts increases. We can see that Expected Infected performs best and out-performs our other proposed heuristic, CASD, with results for other topologies and populations mirroring this result. It can be seen that CASD's performance varies based on the environment, at times out-performing degree discount but not consistently. Therefore, the remainder of our analysis focuses on comparing Expected Infected to the best performing of the existing heuristics, namely Degree Discount.

Expected Infected generally outperforms Degree Discount, although in scale-free environments it suffers. Overall, the difference in performance is larger in small-world networks than in scale-free, as shown by Fig. 1. This is likely due to the difference in connectivity these two network environments present. In a small-world network, most nodes can be reached with just a few hops from any node, but most nodes do not neighbour each other. Scale-free networks, in comparison to small-world, tend to be more connected and nodes are more likely to be direct neighbours. Boosting concepts seem more advantageous in small-world networks, perhaps since they can allow concepts to cascade for extra hops, and in an environment where most nodes can be reached in a few hops this can greatly help with activation numbers. In scale-free networks however, the high density means each node is more likely to have a high degree, making it easier to spread a concept to at least one neighbour.



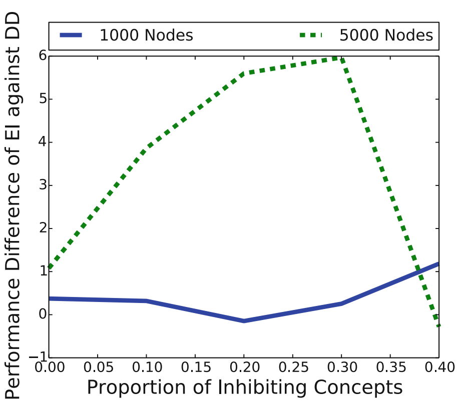
(a) Scale-free network (b) Small-world network

Fig. 2. Difference in activations of Expected Infected (EI) and Degree Discount (DD) against boosting proportions for 1000 and 5000 node graphs

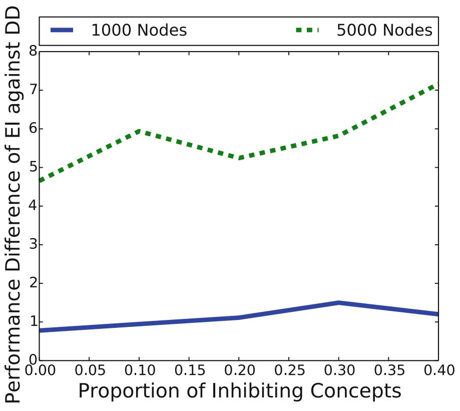
As the proportion of boosting concepts rises, all heuristics improve their total activations, demonstrating the impact of concepts interacting. The advantage of Expected Infected is stable for smaller populations, but is more varied in larger populations. It seems that other network aspects counteract the benefit of targeting boosting concepts. For instance, in larger graphs encountering other concepts may be rarer, making smaller populations more sensitive to concept interaction.

Targeting boosting concepts seems more effective in small-world environments, as their proportion within the network increases. As Fig. 2 shows, there is a small overall decrease in performance difference between Expected Infected and Degree Discount for the smaller scale-free environments. In larger populations there is, overall, a small increase. The results for small-world environments demonstrate a steady performance, increasing slightly at higher proportions for both population sizes. The drop off in performance at high boosting proportions for scale-free networks shows that as boosting concepts become more numerous in this environment, explicitly targeting them becomes less advantageous. Naturally, the more boosting concepts exist the easier it is to encounter them by chance. Furthermore, due to their construction through preferential attachment, scale-free networks often have a core group of nodes with high degree. Both Expected Infected and Degree Discount will target nodes of high degree, and such nodes will be more capable of utilising nearby boosting concepts without explicitly targeting them as the proportion of boosting concepts increases. Small-world networks are less likely to have these central nodes, and so the advantage gained from boosting concepts is more valuable.

It can be seen that the performance of Expected Infected compared to Degree Discount suffers a drop of six activations from 0 to a 0.2 proportion of boosting concepts in larger scale-free networks. Then, from a proportion of 0.2 to 0.3



(a) Scale-free network



(b) Small-world network

Fig. 3. Difference in activations of Expected Infected (EI) and Degree Discount (DD) against inhibiting proportions for 1000 and 5000 node graphs

there is a dramatic rise in the performance difference, with Expected Infected having, on average, 9 more activations. The small drop after this at a proportion of 0.4, considering the previous increase, suggests the existence of an optimum proportion of boosting concepts where explicitly targeting them gives a significant benefit. The environment that creates a particular optimum is not clear. It appears to be a more significant factor within scale-free networks, possibly due to their construction being based around hub nodes. Furthermore, the method of choosing the proportion of boosting and inhibiting concepts for a given single concept may also have an effect. The fairly large standard deviation for the Gaussian distribution, considering there are only 10 concepts within any network, means then number of boosting concepts can be quite varied and at lower mean proportions will often be 0. Future work will explore the effect of changing the standard deviation of the Gaussian distribution to investigate this hypothesis.

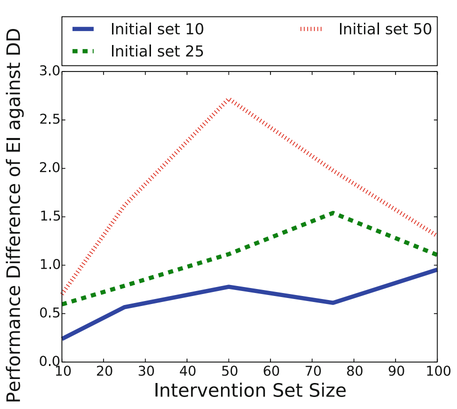
Observing performance against the proportion of inhibiting concepts, Fig. 3 shows that performance tends to increase with more inhibiting concepts. This increase is more pronounced in small-world networks, especially in the larger networks. At a high proportion of inhibiting concepts, it becomes difficult to avoid them by chance. In a heavily inhibiting environment, high degree nodes have a higher chance to encounter inhibiting nodes and consequently have their influence diminished. Part of the effectiveness of Expected Infected appears to be in avoiding inhibiting concepts, rather than in taking advantage of boosting ones, as the highest proportion of inhibiting concepts generally exhibits the best performance difference. Scale-free networks demonstrate this behaviour at lower inhibiting proportions, though at higher levels Expected Infected begins to perform worse compared to degree discount. Once again, this could be due to the hub nodes finding the avoidance of inhibiting concepts impossible.

In many of the environments represented within Figs. 2 and 3 there is a decline in performance when the proportion of boosting or inhibiting concepts reaches 0.2. At this level, it is likely that the boosting/inhibiting concepts are numerous enough that they can be encountered by chance. This will lower the advantage that can be gained by actively targeting or avoiding these concepts. This decline is more prominent in larger networks, likely due to the number of nodes causing the concepts to be sufficiently spread out that they will likely not significantly interact with each other. Together, these factors provide an environment that diminishes the advantages of Expected Infected, hence a drop in performance. Expected infected seems to perform best in two scenarios, namely, when interaction between concepts is low and it can target the advantageous areas that exist or when inhibiting concepts are common and it can actively avoid their detrimental effects.

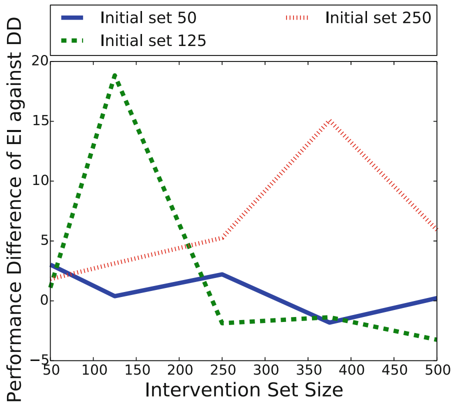
The size of the initial and intervention sets also impacts performance as shown by Fig. 4. For small-world networks we can see in Fig. 4a that larger initial sets results in better performance and, mostly, increasing intervention size also improves the performance over degree discount. This increase in coverage makes concept interaction more likely and the consideration of other concepts more advantageous. Furthermore, as the initial and intervention set sizes increase, Degree Discount will find avoiding inhibiting concepts harder, demonstrating the advantage of avoiding them. In scale-free networks the relationship is less consistent, as the environment around the central nodes likely plays a bigger part. Figure 4b shows that at higher populations, performance often actually decreases as intervention set size increases. With the central nodes likely being targeted by all intervention set sizes and the importance of concept interaction consideration, the extra nodes in bigger intervention sets likely do not have many concepts nearby to take advantage of. This means that degree is more important for these extra nodes, and Expected Infected loses the advantage of boosting and inhibiting nodes. Furthermore, we can see most environments in Fig. 4 have a peak, followed by a harsh decline in performance. In small-world environments the peak happens earlier for larger initial sets, likely a result of the structure of small-world networks. Since most nodes can be reached by any node in a small number of hops, the chance of choosing a node for the intervention set near other concepts increases for all heuristics as the initial set grows. This means that other heuristics gain from concept interaction, lessening Expected Infected's advantage.

A relationship also appears to exist between network density and performance of Expected Infected, since as density increases Expected Infected performs better. The denser a network, the more edges each node has, and avoiding inhibiting nodes by chance becomes less likely, potentially impacting performance. The effect of such network properties will be a key focus in our future work.

Overall the Expected Infected heuristic performs slightly, but consistently, better then Degree Discount in a non-blocking multi-concept environment aside from in some scale-free environments, and out-performs all other heuristics considered. This includes the second proposed heuristic, CASD, which itself



(a) 1000 node small-world network



(b) 5000 node scale-free network

Fig. 4. Difference in activations of Expected Infected (EI) against Degree Discount (DD) for different initial sets against intervention set size

performed inconsistently. CASD occasionally outperforms Degree Discount, particularly in small-world environments or those focused on inhibiting concepts. These are also the environments in which Expected Infected performs best, highlighting that these environments contain key properties for making use of concept interaction. The avoidance of inhibiting concepts seems to be a particular advantage of considering concept interactions, preventing the spread of a concept from being hindered.

7 Conclusion

The study of how ideas, strategies or concepts propagate through a network has many applications. For example, simulations of disease and their infection characteristics can help identify areas at risk of an epidemic that should be the focus of containment, and detecting the influential individuals in a social network allows for the improvement and refining of marketing strategies. This paper introduced an extension of the concept interaction model by Sanz *et al.* [20] to allow for n concepts within the independent cascade. We also proposed two new heuristics, Expected Infected which uses concept relationships to find the expected value of activating a node and Concept Aware Single Discount which adapts the Single Discount heuristic for an environment with concept interactions. Expected Infected was found to out-perform Degree Discount consistently in an interacting concept environment. Specifically, the avoidance of inhibiting factors helps Expected Infected to avoid a concept's spread being hindered. Concept Aware Single Discount was found to be inconsistent, and while it could out-perform Degree Discount in some environments, it is always out-performed by Expected Infected.

Further work to quantify the effect of network properties on concept interactions is needed, to give a better understanding of when best to utilise concept interaction based heuristics. This will include investigating the effect on performance from changing the density of scale-free graphs and the Gaussian function for selecting boosting and inhibiting concept proportions. Observing how our results scale with the increase of concepts within the network would also be of interest, to see if the consideration of inhibiting concepts remains important. The current model is also simplistic in its approach to concept interaction, and extending the model to allow for nodes to deactivate concepts or for concepts to deactivate other concepts could provide more realism to these simulations. If concepts can be deactivated it is likely avoiding inhibiting concepts will become even more important, however the extent to which this is the case remains to be seen.

References

1. Anagnostopoulos, A., Kumar, R., Mahdian, M.: Influence and correlation in social networks. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge discovery and Data Mining, pp. 7–15 (2008)
2. Apt, K.R., Markakis, E.: Diffusion in social networks with competing products. In: Persiano, G. (ed.) SAGT 2011. LNCS, vol. 6982, pp. 212–223. Springer, Heidelberg (2011)
3. Borodin, A., Filmus, Y., Oren, J.: Threshold models for competitive influence in social networks. In: Internet and Network Economics, pp. 539–550 (2010)
4. Brummitt, C.D., Lee, K.-M., Goh, K.-I.: Multiplexity-facilitated cascades in networks. Phys. Rev. E **85**(4), 45–102 (2012)
5. Chakrabarti, D., Wang, Y., Wang, C., Leskovec, J., Faloutsos, C.: Epidemic thresholds in real networks. ACM TISSEC **10**(4), 1 (2008)
6. Chen, W., Wang, Y., Yang, S.: Efficient influence maximization in social networks. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 199–208 (2009)
7. Chen, W., Yuan, Y., Zhang, L.: Scalable influence maximization in social networks under the linear threshold model. In: IEEE 10th International Conference on Data Mining, pp. 88–97 (2010)
8. Domingos, P., Richardson, M.: Mining the network value of customers. In: Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 57–66 (2001)
9. Ferreira, S.C., Castellano, C., Pastor-Satorras, R.: Epidemic thresholds of the susceptible-infected-susceptible model on networks: a comparison of numerical and theoretical results. Phys. Rev. E **86**(4), 41–125 (2012)
10. Goldenberg, J., Libai, B., Muller, E.: Using complex systems analysis to advance marketing theory development. Acad. Mark. Sci. Rev. **2001**(9), 1–20 (2001)
11. Goyal, S., Kearns, M.: Competitive contagion in networks. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing, pp. 759–774 (2012)
12. He, X., Song, G., Chen, W., Jiang, Q.: Influence blocking maximization in social networks under the competitive linear threshold model. In: SIAM International Conference on Data Mining, pp. 463–474 (2012)

13. Karrer, B., Newman, M.: Competing epidemics on complex networks. Phys. Rev. E **84**(3), 36–106 (2011)
14. Kempe, D., Kleinberg, J., Tardos, É.: Maximizing the spread of influence through a social network. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 137–146 (2003)
15. Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., Glance, N.: Cost-effective outbreak detection in networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 420–429 (2007)
16. Pastor-Satorras, R., Vespignani, A.: Epidemic spreading in scale-free networks. Phys. Rev. Lett. **86**(14), 3200 (2001)
17. Pathak, N., Banerjee, A., Srivastava, J.: A generalized linear threshold model for multiple cascades. In: IEEE 10th International Conference on Data Mining, pp. 965–970 (2010)
18. Richardson, M., Domingos, P.: Mining knowledge-sharing sites for viral marketing. In: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 61–70 (2002)
19. Sahneh, F.D., Scoglio, C.: May the best meme win!: new exploration of competitive epidemic spreading over arbitrary multi-layer networks (2013). arXiv:1308.4880
20. Sanz, J., Xia, C.-Y., Meloni, S., Moreno, Y.: Dynamics of interacting diseases (2014). arXiv:1402.4523

Author Index