Discovering Agent Models using Process Mining: Initial Approach and a Case Study

Rob H. Bemthuis* and Sanja Lazarova-Molnar†‡

* Pervasive Systems, University of Twente, Enschede, The Netherlands
r.h.bemthuis@utwente.nl
† Institute AIFB, Karlsruhe Institute of Technology, Karlsruhe, Germany
sanja.lazarova-molnar@kit.edu
‡ Maersk Mc-Kinney Møller Institute, University of Southern Denmark, Odense, Denmark
slmo@mmmi.sdu.dk

Abstract—Agent-based modeling is widely used for modeling and simulation of self-organizing sociotechnical systems that are composed of distributed autonomous agents. In these systems, macro level behaviors emerge from local micro level behaviors of agents that follow rules and interact with each other and the environment. Although the individual agents’ behaviors are typically described by sets of simple rules, the many interactions, heterogeneous populations, and complex topologies can make it challenging, or even impossible, to predict or steer the emergent behaviors beyond micro levels. Hence, the actual behaviors of such systems are generally hard to know beforehand, and they need to be observed to extract realistic models. In this paper, we propose a proof-of-concept approach to discover agents’ underlying models from log data generated from their behaviors, utilizing process mining. To conceptualize and demonstrate our initial approach, we use an illustrative example of the popular Schelling’s model of segregation. Our findings provide encouraging initial evidence on how agent models can be extracted utilizing process mining techniques.

Index Terms—agent-based modeling, agent-based simulation, process mining, Schelling, segregation

I. INTRODUCTION

Agent-based modeling (ABM) has become a well-established methodology that has been applied in many fields of study and to real-world problems [1], [2], spanning a wide spectrum of areas like transportation [3], epidemics [4] and agriculture [5]. ABM can be used to conceptualize realistic self-organizing sociotechnical systems composed of autonomous decision-making entities called agents. Each agent individually determines its situation and generally makes a decision based on a set of simple rules [1], [6]. In the ABM paradigm, macro level behaviors emerge from micro level behaviors of agents interacting with each other and the environment [1]. Even though the rules dictating a system can be relatively simple, resulting emergent behaviors can exhibit complex patterns and provide valuable insights into the dynamics of a real-world system.

Although the rules followed by agents have become more sophisticated in terms of modeling many real-world phenomena [7], [8], learning how to build a realistic agent-based model can be challenging. Effectively dealing with each possible agent motive and relation, and thereby mirroring all processes in the real world, may be impracticable and improbable. For example, we need to improve our understanding of how humans (i.e., agents) make decisions and how these decisions can be formalized in ABM [9]. Real-life complexities, such as implicit assumptions, exceptions, and disruptions, can reveal wrongly assumed modeled rules of behavior.

To this end, ABM is increasingly using empirical data to obtain more realistic model representations [10], [11]. However, the majority of present ABM efforts involve (1) manual development of an agent model, (2) ad-hoc tuning of a large number of parameters [12], and (3) validation limited to a qualitative expert assessment (e.g., surveys or census) or overall fit of aggregate behavior to ground truth using the original data on which the model was calibrated [13]. Little attention has been given to promising data-driven opportunities arising to derive agent behavioral rules through new forms of data and data science [14].

We aim to put forward the possibility to discover agent-based models from data that is typically collected from ubiquitous sources, such as Internet-of-Things (IoT) devices, social media, and other data collection technologies [15]. Purposefully, combining the ABM paradigm with techniques originating from the process mining discipline seems promising to be further explored, especially as agent models can be extracted from data gathered through various (automated) data collection services and devices. Process mining is an analytic discipline aiming to discover, monitor, and improve real processes by extracting knowledge from event logs widely available in all sorts of today’s information systems [16]. This discipline is fundamentally different from traditional modeling approaches, as it starts from observed behavior rather than modeled behavior [17], [18]. As such, process mining provides a means of...
capturing complex processes which have formerly been simplified out of the model [19]. Using process mining with ABM could provide different perspectives for analyzing the macro level behavioral impact of micro level changes [20]. Process mining techniques can connect relevant activities, as recorded in event logs, and analyze the logs from different perspectives by ultimately, turning event data into insights and actions for improving processes (e.g., running more efficiently/effectively), and thus also agent decision-making.

Many processes are producing such digital footprints, including (sociotechnical) systems that can be modeled using ABM. Basically, event logs allow tracing and following objects throughout a system and observing the activities that are attributed/connected to that object. Individual-level details can be observed/derived directly from the digital traces. Furthermore, those footprints may not only be limited to capturing micro level behaviors but could also include elements (e.g., actions and interactions) that span macro level behavior emerging from micro level behaviors [20], [21]. For example, an agent who is interacting with other agents may unravel patterns about other agents because of frequent interactions. Event logs may be able to capture certain behaviors, such as exceptions, which could discover a more realistic and impartial view of the functioning of complex agent-based models. Besides that, as agent-based models are flexible, adaptable, and reconfigurable [22]. ABM can cope intrinsically with complexities arising by comprising data analytics techniques. To some extent, this has already been exemplified by using process mining (e.g., see [20]). Previous work also showed that agents could adapt their decisions based on knowledge derived from discovered process models [23]. Thus, combining ABM and process mining enables a basis to distill agent-based models from digital traces and use these models to discover patterns of emergent behavior phenomena.

While some research has been carried out on combining process mining and ABM, deriving agent-based models (e.g., the decision rules) motivated by the ease with which data is collected nowadays has received scant attention in research literature to our knowledge. In this study, we aim to discover an agent’s underlying model from data recorded from their individual behaviors using a combination of process mining techniques and event log analyses. The proposed approach consolidates ABM and process mining techniques to examine the rules adhered to by individual agents. For illustrating the approach, we consider the classic Schelling model of segregation and aim to show how insights regarding an agent’s model can be given in different scenarios.

The remainder of this paper is structured as follows. In Section II, we recap relevant background knowledge on ABM and process mining, including related work. Section III introduces the ABM approach using a combination of process mining techniques and event log analysis methods. Section IV discusses the case study including illustrative scenarios regarding the discovery of agent rules. Finally, Section V concludes and provides a discussion and pointers for future work.

II. BACKGROUND AND RELATED WORK

First, we provide background on ABM and process mining. Then, since our case study is based on Schelling’s segregation model and we use examples based on this model throughout the paper, we introduce this classic agent-based model. We conclude this section by providing an overview of the existing related work on using process mining for ABM.

A. Agent-based Modeling

An agent can be described as a computational entity (e.g., a software program or a robot) that is situated in some environment and that is capable of perceiving (e.g., sensing) and acting in this environment [24]. There is no general consensus on the agent concept as there are many points of view and an agent could be any class of autonomous entity (e.g., organization, person, system, etc.), encompassing heterogeneous behavior specifications. However, there are some common properties featured by agents, which comprise being autonomous, flexible, self-controlling, interacting (with other autonomous agents), and situated in an environment [24]. Being autonomous implies that an agent’s behavior, at least partially, depends on its own experience.

In the literature, agent-related technologies are known through different terms, such as ABM, agent-based systems (ABSs), multi-agent systems (MASs), and agent-based modeling and simulation (ABMS). ABM refers to the class of computational models used for determining the actions and interactions of autonomous agents (both individual and collective entities) by focusing on assessing their effects on the system as a whole. An agent-based model denotes a computational model describing the actions and interactions of agents in order to understand the system behavior. With an ABS, we refer to an implementation (e.g., in a computer system) in which the key abstraction used is that of an agent. The construct of MAS is also lacking a concise definition. Namely, a MAS is a system composed of several interacting agents [25]. In order to interact successfully, agents require the ability to cooperate, coordinate, and negotiate with each other. In comparison to ABM, MAS is more about the coordination of decisions made by autonomous, deliberative agents that share an environment. A MAS can be modeled using ABM. Similar to [26], with ABMS we refer to the general modeling and simulation paradigm, meaning ABM for the particular task of modeling and agent-based simulation for the execution of the model. It is noteworthy to mention
that agent-based simulation is a special form of discrete-event simulation [27], which can act as a recourse when generating event logs.

B. Schelling’s Segregation Model

Thomas Schelling’s model of segregation [28] is one of the popular agent-based models, illustrating how individual and minor tendencies towards neighbors can lead to a high degree of segregation. In the classic model, agents manifest themselves on a square grid where they can move around and interact with other agents. Each grid cell can contain at most one agent. Agents are of two types, say red and blue, which might represent different races, economic statuses, etc. An agent is happy if a certain proportion of neighboring agents – the ‘threshold of tolerance’ – is of the same color, and unhappy otherwise. At each step, happy agents stay at their current location, while unhappy agents will randomly pick an empty grid cell to move to. The model keeps running until a certain number of steps have been executed or until all agents are happy.

By default, the threshold value is often set to 3 for the centered agent to be happy. That means an agent would be happy with a majority of their neighbors being of a different color. Schelling demonstrated that despite the agent’s willingness to accept a more diverse neighborhood (as indicated through their mild preferences), segregation may occur at a large scale nonetheless from the interactions of different agents. Even weak local preferences can lead to macro phenomena emerging from those interactions. Additionally, these macro level emergent behaviors can feed back onto the individual agents, constraining their choices and behavior.

Variations to the classic model exist, such as having fixed or periodic boundary conditions, including/excluding vacant locations in computing the threshold of tolerance, and varying population and city sizes. Literature reports some examples of model extensions (see e.g., [29], [30]).

C. Process Mining

Process mining covers a broad range of techniques and analysis methods based on event logs by extracting information about processes from these logs [16]. Details about the activities that have been executed can be considered by process mining in the form of event logs [16]. Such digital traces might be scattered over many systems and present in different forms. The practice of process mining attempts to produce an impartial view of how processes are being executed based on data logs about real process execution data. The output can be a process model which can, for example, be used to analyze discrepancies between how processes operate in the real world and the expected behavior.

Process mining assumes the existence of a so-called event log and that it is possible to sequentially record events such that each event refers to an activity and relates to a particular case [16]. An event record describes an activity (a well-defined step in some process) and a case (a process instance). Besides that, event logs may store additional information about events, such as the resource (e.g., person, device, machine, etc.) executing, initiating, or managing the activity, the timestamp of the event, or other data elements recorded with the event [16].

Three types of process mining can be conducted using event logs. Process discovery is about producing a model without using any a-priori information [16]. Conformance checking is about comparing ‘reality’, as recorded in the log, to a model and vice versa. Enhancement, as a third type of process mining, extends or improves an existing process model using information about the actual process recorded in some event log [16]. In this article, we mainly consider process discovery, in which the input is an event log and the output concerns a model such as a process model (e.g., Petri net, BPMN, etc.) or a social network model.

D. Process Mining for Agent-based Modeling

There exists a large volume of published studies on the integrated role of ABM and process mining [20]. Roughly speaking, literature can be divided into two categories. The articles from the first category discuss agent-based models for managing or analyzing business processes, while papers from the second category focus on the generation or verification of agent-based models from (real world) event data [20]. We refer the reader to the work of [20] for research on the former category, whereas we briefly review work on the latter category, as our research aim is to discover agent models from event data.

One of the research papers related to our work is [20], where the authors propose a process mining method grounded in the ABM paradigm called agent system mining (ASM). ASM aims to infer MAS models of operational business processes from event data. Additionally, the authors proposed a framework to map ASM activities and artifacts to different phases and tasks of a MAS modeling lifecycle. Their framework is instrumental and guided us in the development of our approach.

A formalization of how an abstract multi-agent architecture relates to process mining by defining how event logs can be recorded and subsequently analyzed has been provided by [31]. Work of [32] presents a hierarchical Markov model to discover the relationship between micro level and macro level description of a business process. A probabilistic goal recognition problem and discovered models of rational and irrational behaviors of agents have been examined by [33]. Their obtained models can be used to observe behavioral patterns and align observations against process models that are extracted using process mining discovery techniques. Even
so, several existing articles are providing initial analysis on (multi-)agent activity logs and/or support on verification of (multi-agent-based) systems (e.g., [34]–[37]).

While efforts on discovering autonomous aspects of agent behavior by extracting knowledge from event logs are acknowledged, there is a recent call to further develop methods for discovering and enhancing executable agent-based models of business processes [20]. Our main contribution is to provide an initial approach to discover a simple individual micro level agent model from event records using process mining techniques. Our approach consists of a practical course of action, inspired by the ASM framework [20] and the CRISP-DM data science methodology [38], [39]. With a simple individual micro level agent model we refer to obtaining indicators of the (originally) inferred rules of behavior of an agent, which are easily understandable by humans (e.g., parameters of Schelling’s model). Additionally, we extend and specialize the referred methods by (1) using these as a source of reference for designing our approach and (2) gaining concrete knowledge about different aspects of Schelling’s model instantiating the methods. As Schelling’s model is illustrative for many agent-based models, we contend that key characteristics, meanings, and implications of the case could be translated to other (real-world) problems.

### III. AN INITIAL APPROACH TO DISCOVER AGENT MODELS BY APPLYING PROCESS MINING

There are many possible ways to enable ABM such that process mining can be applied. In this section, we first present the assumptions of our proposed approach, followed by our initial approach for the agent model discovery process.

#### A. Assumptions

Our approach for agent model discovery considers design choices and assumptions regarding agents’ behaviors and attributes, as described below.

**Micro level behavior is not known a-priori.** We consider the micro level agent models to be unknown a-priori. It is assumed to not know the decision logic resulting in an agent’s action beforehand. Instead, the approach attempts to discover the micro level agent’s model.

**States and events of agent and environment can be (partly) observed by data logs.** We consider that states and events are observable through data records. This is of importance as process mining techniques require these logs as input. Consequently, we consider that executed activities can be discretized in logical steps (e.g., time or sequence of activities).

ABMs progress from one state change to the next. A state change can happen during a single step executed by an agent. However, it might be that it is not able to capture (the complexity of) all state changes. For example, an agent may first perform negotiations with other agents of which only the final outcome could be observed. An example related to Schelling’s model includes that all unhappy agents are assigned to an empty spot, but that the exact assigned location can depend on the sequence in which the agent’s change request is handled (usually at random).

**Agents make independent step-wise decisions, conditioned on a state.** Each agent makes independent decisions at each step $t$, conditional on state $x$. All features relevant to an agent’s decision are included in $x$. We are presuming to be able to use observed indicators of state $x$ (through the use of data logs), let us call this the observed state $\hat{x}$, for representing state $x$. Next, let $h(x)$ describe an agent’s model (i.e., the rules) based on captured observations. Consequently, we suppose that the “real” model of agent behavior $f(x)$ can be estimated through the extracted model $h(\hat{x})$. Thus, we assume that an agent’s model discovered using the captured log files can accurately represent an agent’s real behavior. This assumption is relatively innocuous as (other) potential features relevant to an agent’s decision could be included by ABM.

Note that agents can be homogeneous or heterogeneous. Heterogeneity can be included in $f$ by considering individual agent characteristics in state $x$, such as personalized preferences (e.g., alternate measures of “happiness”).

**Agents respond to the environment using a static rule set and are devoid of complex learning and/or memory.** Agent-based models are notorious because of the large parameter space that could be facilitated. Additionally, because of the stochastic nature, models often need to be run multiple times and statistical reasoning (e.g., taking averages) is needed to confine to conclusions. Besides that, agent-based models are not (universally) differentiable. This implies that finding a valid agent model representation is not necessarily guaranteed.

To address this challenge, we demarcate our scope by assuming that agents’ actions follow static rules and do not bear any complex learning and/or memory mechanisms. In fact, we consider that the rule set is constant throughout all data records logged in a file. Our approach serves as a first proof-of-concept and disregards the evolution of rules.

#### B. Initial Approach

The approach that we present is inspired by the ASM framework [20] and the CRISP-DM data science methodology [38], [39]. We chose to combine components from these methods as these each has functional parts for our purpose. CRISP-DM is a popular methodology used in the context of data science projects, which consists of six iterative phases and could be combined in a loose implementation with other methodologies. ASM provides a bottom-up approach in the development and
analysis of agent-based models of processes from event logs, including tasks, activities, and artifacts.

Figure 1 presents our approach for discovering agent models from data records generated from their individual behaviors. The approach consists of six phases, each producing artifacts. The phases are mainly based on the CRISP-DM cycle, while the produced artifacts are mainly derived through the ASM framework. In the following, we explain what each phase entails. Please note that although we discuss the phases in a linear and incremental way, it is common to move back and forth between the phases or to perform multiple steps in parallel.

a) Phase 1 (Motivate): postulate motivation for discovering the agent model(s), addressing what problem(s) could be solved or objective(s) could be achieved (e.g., supporting organizational/business decisions) by obtaining an understanding of the working of an agent’s model. Objectives may be formulated in terms of benefits that are expected to be achieved once understanding parts of the underlying agent’s logic. Preferably, people with knowledge of process mining, ABM, and domain experts are involved in this phase.

b) Phase 2 (Plan): select an ABS and make sure that the system is able to produce data that can be translated into the format of an event log. The input may also be an abstract representation of an ABS (e.g., a real-world case that is assumed to be operating like an agent-based model). Define the scope, features, and constraints of the ABS. The first two phases produce a Project Plan and a selection is made regarding an Agent-based System and one or more Candidate Data Logs. The outcomes are preliminary, meaning that revisions might be done based on the next phases.

c) Phase 3 (Understand and Prepare Data): explore and describe the data and check the data quality. Determine (derived) data attributes and their collations. Data quality may be improved by cleaning and filtering data. For this, different methods are possible and are data and model-dependent. This can result in modifications to the earlier phases. Outcomes of this phase include Selected Data Logs and an Data Log Mapping. The latter indicates how data extracted from the ABS is translated into the format of an event record. Note that it can be challenging to select a log such that it is relevant to a single agent, multiple agents, or the environment in a particular application context, which can also be called sub-model discovery [20].

d) Phase 4 (Mining and Analysis): determine which process mining techniques to use, apply those techniques on the data logs, and assess the quality of the mined process model(s) (e.g., fitness). Conducting exploratory techniques may be part of the mining step, including, e.g., social network or control-flow analysis. For example, this can be relevant when the objectives described in the previous phase are more abstract. The exploratory analysis also concerns an analysis of the data log. This phase produces Mined Process Models and Process Insights.

e) Phase 5 (Evaluate): relate the obtained process models to an agent-based model and infer an agent’s logic. Whereas the previous two phases focused on technical data and model assessment, this phase looks at how the obtained process model(s) and insights (e.g., social network) can be explained in terms of imposed agent rules of behavior. Preferably, this should result in an Inferred Agent Model that is understandable to humans (e.g., by using business rules). Obtained process models and insights can be interpreted by a human, but this may also be automated. With interpretation, we mean the process of drawing conclusions from data, evaluating what the results may mean, and communicating about the results.

f) Phase 6 (Use): reflect on the work accomplished and relate this to the identified problems/objectives. Check the phase execution processes and correct or redo anything if needed. This phase determines whether to proceed, iterate further, or initiate a new cycle.

IV. ILLUSTRATIVE CASE STUDY

In this section, we describe how our proposed approach for discovering agent models was applied to Schelling’s model of segregation as a case study. We discuss the activities executed in each phase of the approach, including modeling, analysis results, and other artifacts. For the purpose of this paper, we mainly describe the outcome as obtained after a few iterations across the phases.

a) Motivate: Our main goal is to get insights into agents’ individual rules of behavior in the context of the Schelling’s model of segregation. For a case description of the Schelling’s model, we refer to Subsection II-B. By obtaining a model description of single agents, patterns (if any) of agent dynamics can be discovered. For instance, deviating behaviors (e.g., exceptions) and best strategies may be revealed. In turn, this information can be used to steer agent policy-makers and/or to check whether observed behaviors conform to an agent’s intended behavior. Initially, our goal was: what does an individual agent’s behavior process look like? In subsequent iterations, this objective has been unfolded questions like what is an agent’s tolerance threshold value? The project plan (artifact) is mainly addressed through the present article.

b) Plan: For modeling and simulation, we selected the ABS of segregation dynamics available within the open-source library in Python - AgentPy [40]. In Figure 2, we illustrate the simulation model setup, whereby we assume that not all experiment input parameters (i.e., the tolerance threshold values) are known beforehand (as we aim to discover unknown input values). The model distinguishes between a model step and an agent step to cite:

(i.e., state transition). A model step is initialized at the start of the simulation \((t = 0)\) and increases by 1 once all agent steps are executed. The agent step is called each time an agent executes a step. During both steps, data records are produced. Figure 3 shows examples of possible log files (in the context of Schelling’s model) that may be extracted from the MAS. The figure acts as a template for deriving data record files.

**c) Understand and Prepare Data:** We generated data records during the model setup and each agent step and model step (see Figure 2). Each record stored the following information: agent identification number, position, (first-degree) neighbors, and a sequence counter indicating agent step and model update step. We executed several scenarios to extract data records from the MAS (as shown in Table I). The scope of the data extraction was limited to the results of a single simulation run per scenario. We also preset a number of default parameters, such as: density = 0.95, size = 50 (height and length of the grid), and maximum number of steps = 50.

Table I shows how the agent output was matched to an event log format suitable for process mining. As “timestamp” we considered a sequence counter, which is either an agent step or a model update step (see Figure 2). As “activity”, we specify what the observed state of an agent is (e.g., after relocating it may observe new agents). For this, scenario 1 only considers the number of neighboring agents, while the other two scenarios specify also to which groups the other agents belong.

**d) Mining and Analysis:** We used the Disco miner [41] for discovering process models of the agent behaviors, because of its usability, fidelity, and performance. Disco uses a fuzzy miner, which is useful when having complex log data and simplifying the complexity of the process models in an interactive manner. Figure 4 and Figure 5 show the obtained process models for respectively scenarios 1 and 2. The quality of the mined process models could be considered, but as no guidelines on these metrics for ABM exists to our knowledge, we assumed the main process mining quality dimensions to be sufficient. Please note that for representation purposes, we implemented a processing time per step of one millisecond.

Instead of showing also a process model for scenario 3, we discuss other ways to find indicators about the agent rules: most frequently occurring states (Table II), traces of single agents (Table III), the average happiness perceived based on the number of neighbors (Figure 6), and if-else statements based on the first degrees neighbors (Figure 7). In the next phase we will interpret the findings. Please notice that these process insights are obtained by inspecting only the obtained data records (of scenario 3). Our suggestions to extract agent rules are preliminary and further research is needed to explore not only how to determine which process insights are relevant from the data records but also how process mining techniques can make the rule extraction ways more rigorous.

**e) Evaluate:** After inspecting the process model for scenario 1, it may be observed that the habitat of many agents seems relatively densely populated. Indicators for this are the high number of absolute frequencies on states 7 and 8. At the initialization of the model, there were respectively 720 and 1351 cases known with 7 and 8 neighbors. These numbers increased substantially to 1121 and 1864, indicating that many agents might be relocating to densely populated areas. In Figure 5 and Table II, we discover similar findings. These findings provide initial insights into the processes of the agents. Next, we discuss several ways of extracting agent rules.

We first, discuss an approach based on individual agent traces. For this, we consider the traces as shown in Table III. Rules can be extracted in several ways. We provide initial insights into the processes of the agents. Next, we discuss several ways of extracting agent rules.

![Fig. 1. Conceptual representation of the approach](image-url)
the agents are happy or unhappy. Regarding the first one, the minimum number to be happy is 5 and the maximum is 8, while the minimum number to be unhappy is 7 and the maximum is 8. The following rule can be extracted: “if there are 5 or 6 neighbors, then the agent is happy, if there are more neighbors, the agent is either happy or unhappy”. Regarding the latter one, the minimum and maximum percentage of similar agents in order to be happy is respectively 37.5% and 100%, for unhappy this is 0% and 28.6%. As rules, we extract this: “if the number of similar neighbors is higher than or equal to 37.5%,
then the agent is happy; if this percentage is lower than or equal to 28.6%, then the agent is unhappy; if this percentage is in-between 28.6% and 37.5%, then the agent is either happy or unhappy”. These example rules serve as an initial proof of concept and further analysis is needed regarding the discovery of more advanced rules. This forms part of our future research.

Second, we describe a probabilistic way to look at agent rules. Figure 6 shows the average happiness of an agent per number of neighboring agents. We calculated the average happiness scores of all data records. Agent rules can be extracted directly from the graph, such as: “if there are 5 direct neighbors, then there is a 95.4% chance of the centered agent being happy” or “if there are 18 second degrees neighbors (95.7% chance of being happy) and 30 third degrees neighbors (95.5% chance of

to cite:
being happy), then the chance of the centered agent to be happy is $(95.7\% \times 95.5\%) = 91.5\%$.

Third, we listed the happiness levels given a specification of the first degrees neighbors (i.e., number of agents and similar agents) (Figure 7). As we considered the entire data set, the results indicate having a homogeneous agent population (i.e., all agents following similar rules). This figure can be used as a lookup table for the extracting agent rules. For example, "if there are 7 neighbors and at least 3 similar neighbors, then the centered agent is happy".

Notice that the presented ways to extract business rules are preliminary and primarily driven by inspecting the data records. As also mentioned by [20], we call for research on metamodels, formalisms, and techniques suitable for representing and analyzing (heterogenous) agent models discovered from process data.

f) Use: The implementation of this phase is still ongoing. For example, there are future plans to conduct more sophisticated analysis on event logs generated by a MAS. In particular, future work would focus on discovering more sophisticated agent rules (e.g., stochastic, step-wise, or evolving rules). We also consider automating parts of the model extraction phases, which can be useful when analyzing large-scale agent-based systems and models.

V. CONCLUSION AND DISCUSSION

In this paper, we presented an initial approach to discover an agent’s underlying model from data recorded from their individual behaviors using agent-based models and process mining. The approach consists of data extraction and process mining techniques, by utilizing produced data logs. The approach is demonstrated on the Schelling’s model of segregation. In the case study, we observed emergent behavior patterns that might be linked to the originally imposed agent models. The findings suggest that our approach can provide preliminary insights into how the rules of behaviors of individual agents can be discovered.

Our approach has some limitations. The actual rules of behavior followed by agents can be much more complex. Nevertheless, our study provides some evidence that process mining techniques are a valuable means for discovering agent models by examining traces of emergent behavior. The scope of this study was also limited in terms of data log complexity. We mapped codified ABM features to the format of an event log. However, data records may be much more diverse, constrained, and even nonexistent because of, e.g., privacy and security reasons. Similarly, the scope of this research was limited in terms of evaluated distributed/decentralized intelligence featured by agents.

This work’s natural progression is to implement more realistic and real-life demonstrations to evaluate if and how our approach enhances the usability of process mining for ABM. Another future research direction is to explore the interaction between agents and the environment where agents could adapt and learn from their behaviors. It would be interesting to explore the wide variety of alternative ABS designs, including, e.g., relaxing assumptions about the knowledge available to agents, agent heterogeneity, and network topology. Lastly, further work could examine more closely the links between algorithmic designs of process mining techniques and ABM. It is our intention that this presentation and discussion, although preliminary, will expedite the design and

---

**to cite:**

testing of ABM integrating process mining techniques for future projects.

ACKNOWLEDGMENT
This work was partly supported by the Netherlands Organization for Scientific Research (NWO) under the DataRel project (grant 628.009.015). R.H. Bemthuis has also received funding from the European Union’s Erasmus+ programme.

REFERENCES