Formalization of Emergence in Multi-agent Systems

Yong Meng Teo and Ba Linh Luong
Department of Computer Science
National University of Singapore
13 Computing Drive
Singapore 117417
{teoym,luongbal}@comp.nus.edu.sg

Claudia Szabo
Department of Computer Science
The University of Adelaide
North Terrace
Adelaide 5005, Australia
claudia.szabo@adelaide.edu.au

ABSTRACT

Emergence is a distinguishing feature in systems, especially when complexity grows with the number of components, interactions, and connectivity. There is immense interest in emergence, and a plethora of definitions from philosophy to sciences. Despite this, there is a lack of consensus on the definition of emergence and this hinders the development of a formal approach to understand and predict emergent behavior in multi-agent systems. This paper proposes a grammar-based set-theoretic approach to formalize and verify the existence and extent of emergence without prior knowledge or definition of emergent properties. Our approach is based on weak (basic) emergence that is both generated and autonomous from the underlying agents. In contrast with current work, our approach has two main advantages. By focusing only on system interactions of interest and feasible combinations of individual agent behavior, state-space explosion is reduced. In formalizing emergence, our extended grammar is designed to model agents of diverse types, mobile agents, and open systems. Theoretical and experimental studies using the boids model demonstrate the complexity of our formal approach.

Categories and Subject Descriptors

D.2.4 [**Software Engineering**]: Software/Program Verification—*validation*; I.6.5 [**Simulation and Modeling**]: Model Development—*modeling methodologies*

General Terms

Verification, Theory

Keywords

Emergent behavior, multi-agent system, simulation

1. INTRODUCTION

Systems with a large number of components and complex interconnection are ubiquitous [33, 40]. If components interact with other components and make decisions individually based on internal logic, the system can be modeled as a multi-agent system where agents play the role of components [19]. A system is said to be complex if it exhibits designed (expected) properties that can be derived from the system specification, as well as properties that are irreducible from knowledge of the interconnected components [12, 13, 23]. These irreducible properties are defined as *emergent properties* or *emergence*. For example, flocking is an emergent property of a group of birds [33]. Having attributes and behavior rules of the birds does not enable us to identify the flocking patterns or how they avoid obstacles.

Originating from philosophy [6, 20, 30], emergence has become of interest for studying complex systems and multi-agent systems. Emergence study provides great potential to understand the interactions of agents and their environment [9, 10], and to explain how natural systems work [35]. Furthermore, it is important to know whether emergent properties are beneficial or harmful so that we can exploit the positive impacts of the beneficial properties and minimize, or even prevent negative consequences due to the harmful properties. On the one hand, emergence is beneficial because it confers additional functionalities to the systems. For instance, users adapt software products to support tasks that the designer never intended. On the other hand, emergence may be harmful, such as unforeseeable failures make a system harder to design, analyze, and control [27]. As a result, emergence in a system should be predicted and verified. Unfortunately, it is difficult to *predict* emergence [34]. According to Dyson [14], emergent phenomena cannot be predicted through analysis at any level simpler than that of the system as a whole. Even worse, *verifying* that emergence exists is also difficult [32]. Although in some cases, emergent properties can be regular, which is recurring and recognizable, it does not imply that they are easily recognized [20]. Most studies focus only on post-mortem observation of emergence when a tangible representation of the system is available to examine [17, 25], rather than on verifying that emergence exists.

Kubik [22] proposed a method for formalizing weak emergence without prior knowledge of emergence. The approach adopts a set-based view on the world, where emergence is seen as the difference between a calculated and an observed behavior. However, like other work, the definition of emergence is too broad because emergence is any kind of system states that cannot be obtained from summing individual agents' behaviors. Moreover, the calculation of the expected behavior suffers from state explosion and renders the

SIGSIM-PADS'13, May 19–22, 2013, Montréal, Québec, Canada.

method infeasible for practical use [38, 39]. Our contribution is twofold. First, based on Kubik's approach, we propose a grammar-based framework with a new definition of emergence that distinguishes possible system states and emergent properties of interest, from the larger set of all combinations of system states. Second, in contrast to Kubik's approach, our grammar allows for different agent types, includes mobile agents, and permits the study of open systems where the number of agents varies over time.

The remainder of the paper is organized as follows. Section 2 reviews different perspectives, classifications, and formal studies of emergence. Our emergence definition and proposed approach are presented in Section 3. Using the boids model [33], we show how to apply our approach to detect flocking behavior in a group of birds. Section 4 presents a theoretical and experimental analysis of our work. Section 5 concludes this paper and discusses future work.

2. RELATED WORK

Despite a long history of complex systems research [28], there is no consensus on the definition of emergence. Emergence can be considered in three main perspectives, namely, philosophy, natural and social sciences, and computer science. From a *philosophical* perspective, emergent properties are subjectively a product of both the unexpected behavior of complex systems and the *limitations* of the *observer's knowledge* [21] and the tools employed. Other limitations come from the scale and level of abstraction under which the system is observed [7]. *Scientific perspectives* [1, 22] criticize the idea of temporary lack of knowledge of the observer because it implies that we are unable to study emergence scientifically. Therefore, emergent properties are defined as intrinsic to a system and independent from the eye of the beholder [11]. *Natural and social sciences* explain emergence via theories, mainly in physics, biology, chemistry, and human behavior. Two concepts of emergence in this perspective are self-organization and hierarchy. First, emergence is the formation of order from disorder based on *self-organization* [16]. Second, a property is emergent if it is discontinuous from the properties of the components at the lower levels in the *hierarchy* [2]. *Computer science* aims to predict, verify, validate, and reason about emergence. Prediction is done before the observation of emergence, while verification is done only when emergence is observed. Verification of emergent properties is a complex task because it may lead to combinatorial explosion in terms of the system states. More importantly, emergent properties should be defined first before verifying which is typically not straightforward. Verifying emergence without prior knowledge of emergence is a research challenge. Validation allows us to determine whether an emergent property is beneficial or harmful[37]. Finally, reasoning enables the understanding of the cause-and-effect of emergence. The most frequent studied type of emergence in computer science is *weak* (or *basic*) emergence [4, 6, 8, 16, 22].

A macro level property characterizing the system as a whole is defined to be *weakly* emergent if it can be derived from the micro level dynamics but only by a finitely long simulation [5]. Other definitions see weak emergence in the case whereby the whole is constrained or influenced by the parts through *upward causation* (UC), but at the same time the parts are influenced by the whole through *downward causation* (DC) [6, 30]. DC can be either positive or negative. First, positive DC amplifies UC and leads the system to unstable states [31]. For example, when the general price level of goods and services raises, the cost of living increases. Higher living cost demands higher income, that in turn results in higher prices of goods and services. The cycle continues until a policy to decrease the inflation rate is introduced. Second, negative DC weakens the UC and keeps the system in stable states. There is a balance between autonomy through UC and self-organization through DC. Birds fly in different directions to avoid collision. At the same time, they stay close to each other, try to match the neighbors' speed, and steer to the center of the flock.

Scientific study of emergence requires a formal specification of the system, system components, and even emergent properties. Emergence formalisms can be mainly classified into three categories, namely, variable-based [15, 26, 35], event-based [10], and grammar-based [22]. Each has advantages and disadvantages. In *variable-based* methods, a variable is chosen to represent emergence [29]. Changes in the center of mass of a group of birds, for example, can be used for identifying flocking behavior. Usually, emergence is measured numerically, i.e. quantitative emergence, based on information theory and probability theory. Many variable-based studies deploy Shannon entropy [36] to measure the uncertainty and unpredictability of a system with regard to a particular attribute [15, 26]. The more structural a system is, the lower entropy it has. Entropy is, however, not suitable for systems with continuous attributes. Although differential (continuous) entropy can be defined [31], it lacks some important characteristics, and usually is not a good measure of uncertainty. For example, the differential entropy can be negative because probability density functions could be greater than one; also it is not invariant under continuous linear transformations of variables [31]. Fisch et al. [15] regard emergence as an unexpected or unpredictable change of the distribution underlying the observed samples. However, the computation is costly and user intervention is required. Seth [35] defines G-emergence as a measure of emergence based on G-causality and G-autonomy, which compute the dependence and the autonomy of a variable with respect to a set of other variables respectively. However, a set of variables must be defined and the computations of G-causality, G-autonomy, and G-emergence are expensive. Variable-based approach limits describing details of components, so it gives little understanding of the causes of emergent properties. In addition, defining the abstraction level at which the variable is measured is non-trivial.

In *event-based* methods, component behavior is defined as a sequence of events at some level of abstraction [10]. A *simple event* is a state transition that results from the execution of a single state transition rule. A *complex event* is either a simple event or two complex events satisfying a set of constraints with respect to each other. Emergence is complex events that can be reduced to a sequence of simple events. As with the variable-based approach, event types and emergent behavior have to be formalized beforehand. Identifying and defining the proxy event is not trivial.

To overcome the challenging problem of a prior definition of emergence, Kubik [22] proposes a *grammar-based* approach. An extended cooperating array grammar system is used to formalize the environment, agents, and cooperation among them. Behavior of the system is therefore the language, i.e. a set of words, resulting from derivations of

grammars (agents) on the tape (two-dimensional array of symbols). The approach is based on an intuition of emergence that "the whole is more than the sum of its parts". An emergent property is defined as a system state that results from the interactions of agents and cannot be produced by summing their individual states. Given an initial system state, two languages L_{WHOLE} and L_{PARTS} are calculated, using two grammar systems that define the execution of agents together and individually respectively. L_{WHOLE} defines a set of system states produced by the system agents acting together and interacting as a whole. L_{PARTS}, on the other hand, presents the sum of the agents' behaviors without considering the interactions among them. As a result, emergence is defined as $L_{WHOLE} - L_{PARTS}$. The approach is exemplified using a small game of life example. The grammar-based approach does not require a priori definition of emergence. This approach, however, has four main limitations: (1) there is no explicit provision for agent type, (2) it is unclear how to deal with mobile agents, (3) in the game of life example, agents are stationary, the number of agents is always equal to the number of cells, and (4) the state spaces of L_{WHOLE} and L_{PARTS} are unnecessarily large.

3. PROPOSED APPROACH

Our approach for the verification of emergent properties extends Kubik's grammar-based approach [22] to derive emergent property states. The set of emergent property states is defined as the difference:

$$L_\xi = L_{whole} - L_{sum} \tag{1}$$

where L_{whole} describes all possible system states due to agent-to-agent and agent-environment interaction, and L_{sum} is the sum of all individual agents' behaviors, without considering agent interactions. L_ξ contains the set of system states that are in L_{whole} but not in L_{sum}. This broad perspective of emergence, as in Kubik's approach, leads to state explosion when determining L_{sum} and L_{whole}. This is because all possible combinations of individual agent states are considered following a defined superimposition operator, without including system-defined rules.

We propose a new perspective that significantly reduces the state space for L_ξ. Firstly, it is important to highlight here that while Kubik refers to the difference $L_{whole} - L_{sum}$ as emergence, we refer to this set as the emergent states set, because it is the set from which emergent properties can be deduced. Secondly, we observe that the size of L_{whole} is dependent of the number of interactions and state transition rules defined by a modeler, and a subset of interest (to the modeler) from all the rules in a given system. This would be the case, for example, when we consider different kinds of rules in the modeling a flock of birds: the entire rule set, or some rules of interest while ignoring others. As shown in Figure 1, L_{whole} can be redefined as:

$$L_{whole} = L^I_{whole} \cup L^{NI}_{whole} \tag{2}$$

where L^I_{whole} is bounded by the number of interaction rules that are of interest to the user for the particular study, and L^{NI}_{whole} represents the set of all possible system states that are not of interest.

The size of L_{sum} increases exponentially with the increase in the number of agents as all possible combinations of agent states are considered. Similarly, L_{sum} can be also redefined as follows:

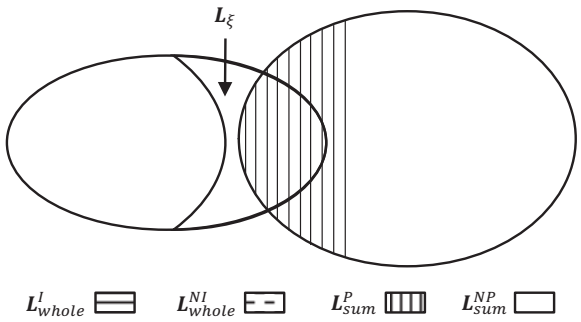

Figure 1: Set of Emergent Property States

$$L_{sum} = L^P_{sum} \cup L^{NP}_{sum} \tag{3}$$

Although it is mathematically possible to derive L_{sum}, in practice there is a subset of feasible combinations of agents' behaviors (L^P_{sum}), and the remaining are combinations of agents' behaviors that will not exist in practice (L^{NP}_{sum}).

Given the above, the set of emergent property states L_ξ is redefined as:

$$L_\xi = L^I_{whole} - L^P_{sum} \tag{4}$$

Emergent property states, with respect to a model of interest, are due to non-trivial interactions among agents, and cannot be derived by summing individual agents' states. As shown in Figure 1, there are states in L^I_{whole} that can be found in L^P_{sum}, in other words,

$$L^I_{whole} \cap L^P_{sum} \neq \emptyset \tag{5}$$

Intuitively, these states are resultant from agent computation that do not require interactions, or from the interactions of agents that has no effect on agent behavior. For example, in the flock of birds model detailed in Section 3.2, two individuals can be very far apart and as such interaction rules between them have no effect.

Contrary to Kubik's approach, which regards emergent properties as system states, we do not consider that L_ξ contains the emergent behavior but only states that together, following particular criteria, can form an emergent property. In the next step, we propose to determine whether L_ξ contains emergent properties that have been seen before, or that are beneficial or harmful to the system. Towards this, we propose to use emergent properties criteria derived from the system expert, as well as information obtained from L^{NI}_{whole}. In this paper, we show the process of calculating the set of emergent property states L_ξ.

Next, we enhance the grammar-based approach with three main extensions: (1) introduce agent type (A_{ij} denotes an agent instance j of type i), (2) introduce mobile agents by defining mobility as attributes of agents $P_i = P_{i\_mobile} \cup P_{i\_others}$ where P_i denotes the set of attributes of agents of type i, (3) model open systems where agents can enter and leave system. As a result, the proposed approach has a wide variety of applications in which components belong to different types, can move in space, and have multiple attributes. These kinds of systems are ubiquitous in practice such as traffic networks [24], social networks [18], and flock of birds [33], among others. Our proposed approach consists of two major steps, namely, modeling the system as a multi-agent system, and the application of our extended grammar-based formalism for different types of mobile agents.

3.1 Emergence Formalism

A multi-agent system consists of m different types of n agents (A) interacting in an environment (E). The environment E is a part of the system that lies outside the agents and can be regarded as a platform for agent interactions. For simplicity, we assume that E has no behavior rules and changes only as a result of agent actions. Changes of the environment, in turn, impact the agent behavior. E is modeled as a 2D grid[1] that is subdivided into c units called cells (e). Changes of the environment are therefore changes of the states of the cells. For example, a cell turns from "occupied" to "free" when the agent that occupies the cell moves to another cell. V_e denotes the set of possible states of the cell e. Similarly, V_E denotes the set of possible cell states. In addition, the environment state is made up of the states of all cells in the environment. $s_e(t)$ and $S_E(t)$ denote the state of cell e and the environment E at time t respectively. Table 1 provides a glossary of notations used for formalizing the system. Agents are classified into m different types in which all agents of the same type have the same attributes and behavior rules. A_{ij} denotes an agent of type i ($1 \leq i \leq m$) and instance j ($1 \leq j \leq n_i$), where n_i is the number of agent instances of type i, and $n_1 + n_2 + ... + n_m = n$. An agent of type i is characterized by three factors: a set of attributes for agents of type i (P_i), a set of behavior rules for agents of type i (R_i), and an initial state ($s_{ij}(0)$). It is important to distinguish agent attributes from their values. For instance, attribute color has several values, such as red, yellow, and green. Only the initial state $s_{ij}(0)$ is specific to the agent, while P_i and R_i are common to all agents of type i. P_i consists of two main subsets: P_{i_mobile} modeling the mobility of agents of type i and P_{i_others} modeling other attributes.

Agent behavior (L) is characterized by behavior rules (R) that define how agents interact with other agents and the environment. We assume that no evolutionary processes are involved in the system, i.e. behavior rules do not change over time. R_i denotes the set of behavior rules of agents of type i. R_i consists of R_{i_mobile} that impacts agent mobility, i.e. changes values of attributes of P_{i_mobile}, and R_{i_others} for the rest. An agent changes its state because it is triggered by a rule that affects the agent itself or one of its neighbors changes state. The state of A_{ij} at time t, denoted by $s_{ij}(t)$, is defined by values of its attributes at time t. V_{A_i} denotes the set of possible agent states for agents of type i.

3.1.1 System Formalism

A multi-agent system consisting of an environment and a set of agents is formalized as an extended cooperating array grammar system where context-free grammars represent agents, and a two-dimensional array of symbols represents the environment. Each grammar has its own rewriting rules defining how the grammar cooperates with the other grammars and with the array, i.e. rewrite the symbols on the array. A system of m agent types and a total of n agents $A_{11}, \ldots, A_{1n_1}, \ldots, A_{mn_m}$ interacting in a 2D grid environment of c cells is defined as follows:

$$GBS = (V_A, V_E, A_{11}, \ldots, A_{1n_1}, \ldots, A_{mn_m}, S(0)) \quad (6)$$

where V_A denotes the set of possible agent states for all agent types, V_E denotes the set of possible cell states, A_{ij} denotes an agent of type i ($1 \leq i \leq m$) and instance j ($1 \leq j \leq n_i$), and $S(0)$ denotes the initial system state. A system state is composed of the state of the environment and the states of all agents.

For the environment (E),

$$V_E = \bigcup_{e=1}^{c} V_e \quad (7)$$

where V_e denotes the set of possible states of cell e. The state of the entire environment is made up of the states of all its cells. The states of cell e and the environment E at time t are $s_e(t) \in V_e$ and $S_E(t) \in V_E^c$ respectively.

For the agents (A),

$$V_A = \bigcup_{i=1}^{m} V_{A_i} \quad (8)$$

where V_{A_i} denotes the set of possible states for agents of type i.

Agent of type i ($1 \leq i \leq m$) and instance j ($1 \leq j \leq n_i$), A_{ij}, is defined as follows:

$$A_{ij} = (P_i, R_i, s_{ij}(0)) \quad (9)$$

where P_i denotes the set of attributes for agents of type i, R_i denotes the set of behavior rules for agents of type i, and $s_{ij}(0)$ denotes the initial state of the agent. P_i is defined as:

$P_i = P_{i\_mobile} \bigcup P_{i\_others}$ where

$P_{i\_mobile} = \{\text{x} \mid \text{x is an attribute that models mobility}\}$

$P_{i\_others} = P_i \setminus P_{i\_mobile}$

Agents change their states according to behavior rules that are regarded as functions from a set of agent states to the set itself. Some of these rules that affect the mobility of agents, for example change location or speed, are defined as mobile rules.

$R_i : V_{A_i} \rightarrow V_{A_i}$

$R_i = R_{i\_mobile} \bigcup R_{i\_others}$

A_{ij} has an initial state $s_{ij}(0) \in V_{A_i}$. The system state at time t ($S(t)$), is composed of state of the environment ($S_E(t)$) and states of agents ($s_{ij}(t)$) at time t. Hence,

$$S(t) = S_E(t) \bigcup_{\forall i \forall j} s_{ij}(t) \quad (10)$$

3.1.2 Verification of Emergence

Our approach for the verification of emergence computes two sets of system states corresponding to the two levels of abstraction defined above, namely, the macro level when regarding the system as a whole and the micro level when regarding the system as an aggregation of its individual agents. The set of emergent system states is defined as:

$$L_\xi = L^I_{whole} - L^P_{sum}$$

L^P_{sum} can be determined by adding agent constraints among others but is not straightforward. We leave this as future work and for simplicity we use L_{sum} for the rest of this paper.

Taking the interactions of agents (denoted as $GROUP$) into account, the system behavior of interest (L^I_{whole}) returns a set of words (w) that represents the set of system

[1]E can be easily extended to model other topologies.

	Notation	Description
System	$S(t)$	state of system at time t
Environment	V_E	set of possible cell states
	V_e	set of possible states of cell e
	$S_E(t)$	state of environment at time t
	$s_e(t)$	state of cell e at time t
Agent Type	m	number of agent types
	n_i	number of agents of type i $(1 \leq i \leq m)$
	V_{A_i}	set of possible states for agents of type i
Agent	n	number of agents
	A_{ij}	agent of type i $(1 \leq i \leq m)$ and instance j $(1 \leq j \leq n_i)$
	V_A	set of possible agent states for all agent types
	P_i	set of attributes for agents of type i
	R_i	set of behavior rules for agents of type i
	$s_{ij}(t)$	state of agent A_{ij} at time t
	$L_(A_{ij})$	set of system states representing the behavior of agent A_{ij}
Emergence	$\oplus$	superimpose operator
	L^I_{whole}	set of system states representing behaviors of interest due to agent interactions
	L^P_{sum}	set of possible system states representing all combinations of agents' behaviors
	L_ξ	set of emergent property states

Table 1: Glossary of Notations

states reachable from the initial system state. Given an initial state, the system is simulated until it reaches a state that has already appeared before. We stop the simulation when the system state repeats itself. Complex adaptive and nonlinear systems with strange attractors where system states vary continuously are not considered in this study. L^I_{whole} with respect to the initial state is therefore the set of all distinct states obtained as follows:

$$L^I_{whole} = \{w \in V^{c+n} \mid S(0) \Rightarrow^*_{GROUP} w\} \quad (11)$$

The sum of agents' behaviors (L_{sum}) is defined as the set of words resulting from superimposing behaviors of individual agents.

$$L_{sum} = \oplus(L(A_{11}), \ldots, L(A_{1n_1}), \ldots, L(A_{mn_m})) \quad (12)$$

where $L(A_{ij})$ denotes the behavior of agent A_{ij} and $\oplus$ is the superimpose operator defined in [22]. The superimpose operator $\oplus$ does a sum of the individual agents' behaviors when they do not interact with each other. Let $w_1 = a_1 a_2 \ldots a_x, w_2 = b_1 b_2 \ldots b_y$ be words of symbols $a_i, b_j, 1 \leq i \leq x, 1 \leq j \leq y$ over an alphabet $(V_A \cup V_E)^+$, and ϵ denotes the empty symbol. Hence, the superimposition of the word w_1 on the word w_2 is a function $\oplus : V^* \times V^* \rightarrow V^*$ that results in $w_{supimp} = c_1 c_2 \ldots c_z$ of symbols $c_k, 1 \leq k \leq z$, defined as follows:

1. $z = max(x, y)$;
2. if $a_i \in V_A$ then $c_k = a_i$;
3. if $a_i = \epsilon$ then $c_k = b_j$;
4. if $b_j = \epsilon$ then $c_k = a_i$;
5. if $a_i \in V_E$ and $b_i \in V_E$ then $c_k = a_i$;
6. if $a_i \in V_E$ and $b_j \in V_A$ then $c_k = b_j$.

The superimposition is done over all permutations of n behaviors of agents. It is important to note that ordering is important in the process of calculating the superimposition. The expression below shows the superimposition of three languages L_1, L_2, and L_3.

$$\begin{aligned}\oplus(L_1, L_2, L_3) = {} & L_1 \oplus (L_2 \oplus (L_3)) \cup L_1 \oplus (L_3 \oplus (L_2)) \\ & \cup L_2 \oplus (L_1 \oplus (L_3)) \cup L_2 \oplus (L_3 \oplus (L_1)) \\ & \cup L_3 \oplus (L_1 \oplus (L_2)) \cup L_3 \oplus (L_2 \oplus (L_1))\end{aligned}$$

Defining the sum of the individual agents' behaviors is difficult. Ideally, the result should contain exactly all designed system states that can be derived from the system specification. Unfortunately, this is only true if agents are independent from the others in the system. Given the initial system state, we obtain n states where each consists of one agent. Considering only the agent in the system, the agent's behavior returns a set of words (w) that represents the set of system states reachable from the corresponding system state. $L(A_{ij})$ is defined as follows:

$$L(A_{ij}) = \{w \in V^{c+1} \mid (S_E(0) \cup s_{ij}(0)) \Rightarrow^* w\} \quad (13)$$

Agent symbols have priority over environmental symbols. Any non-empty symbol has priority over the empty symbol ϵ. For example, consider $V_A = \{a_1, a_2, a_3\}, V_E = \{o, f\}$, $L_1 = \{fa_1of\}, L_2 = \{oa_2ff\}, L_3 = \{foffa_3\}$. Then the superimposition of the agents' behaviors will be the language $L_{sum} = \{fa_1ofa_3, oa_2ffa_3, fa_1ffa_3, fa_2ffa_3\}$.

3.2 Example: Flock of Birds Model

The boids model [33] captures the motion of bird flocking and is a seminal example for studying emergence [9]. At the macro level, a group of birds tends to move in a V-like formation, which has aerodynamic advantages, obstacle avoidance, and predator protection, regardless of the initial positions of the birds. At the micro level, three simple rules define how each bird flies: (1) separation - steer to avoid crowding neighbors, (2) alignment - steer towards the average heading of neighbors, and (3) cohesion - steer towards the average position of neighbors [33].

To demonstrate our proposed approach, we extended the boids model to include two types of birds, ducks and geese. For ease of discussion, we model a multi-agent system with

ten birds with equal numbers of ducks and geese interacting in an environment represented as a 2D grid of 8×8 cells. Each cell is *occupied* by a bird or *free*, and two birds cannot be located at the same cell at the same time. Birds have two attributes: position and velocity. Position and velocity model birds' mobility. The *position* of a bird is location of the cell occupied. The *velocity* is a vector that contains moving direction and speed. Ducks can fly zero, one, or two cells per time step in one of eight directions: north, north-east, east, south-east, south, south-west, west, and north-west. Similarly, geese can fly at the maximum speed of three cells per time step. The vector representation for velocity of ducks is shown in Table 2. Birds behave according to three

Direction	Speed		
	0	1	2
North	(0,0)	(0,1)	(0,2)
North-East	(0,0)	(1,1)	(1,2), (2,1), (2,2)
East	(0,0)	(1,0)	(2,0)
South-East	(0,0)	(1,-1)	(1,-2) (2,-1), (2,-2)
South	(0,0)	(0,-1)	(0,-2)
South-West	(0,0)	(-1,-1)	(-1,-2), (-2,-1), (-2,-2)
West	(0,0)	(-1,0)	(-2,0)
North-West	(0,0)	(-1,1)	(-1,2), (-2,1), (-2,2)

Table 2: Vector Representation for Velocity of Ducks

rules: (1) separation: avoid collision with nearby birds, (2) alignment: fly as fast as nearby birds of the same type, and (3) cohesion: stay close to nearby birds of the same type.

3.2.1 System Formalism

The boids model is formalized as:

$$GBS_{boid} = (V_A, V_E, A_{11}, \ldots, A_{15}, A_{21}, \ldots, A_{25}, S(0))$$

where A_{1j} denotes a duck instance j ($1 \leq j \leq 5$), and A_{2j} denotes a goose instance j ($1 \leq j \leq 5$), $V_A = V_{A_1} \cup V_{A_2}$ denotes the set of possible states for the ducks (V_{A_1}) and the geese (V_{A_2}), V_E denotes the set of possible cell states. $S(t) \in (V_A \cup V_E)^+$ denotes system state at time t, and $S(0)$ denotes the initial system state at time zero.

For cell e, $V_e = \{o, f\}$ where o means occupied and f means free, and $s_e(t) \in V_e$. For the entire environment E, $V_E = \bigcup_{e=1}^{64} V_e = \{o, f\}$, and $S_E(t) \in V_E^{64}$, where 64 = 8×8 is the number of cells.

A duck instance $A_{1j}(1 \leq j \leq 5)$ is defined as follows:

$$A_{1j} = (P_1, R_1, s_{1j}(0))$$

where

$P_1 = P_{1\_mobile} \cup P_{1\_others}$

$P_{1\_mobile} = \{position(g_{1j}), velocity(v_{1j})\}$

$P_{1\_others} = \varnothing$

$V_{A_1} = \{(x, y) | 1 \leq x \leq 8, 1 \leq y \leq 8\} \times \{(\alpha, \beta) | -2 \leq \alpha \leq 2, -2 \leq \beta \leq 2\}$

$R_1 = R_{1\_mobile} \cup R_{1\_others}, R_{1\_others} = \varnothing$

$s_{1j}(t) \in V_{A_1}$

R_{1_mobile} defines the update of the position $g_{1j}(t)$ and the velocity $v_{1j}(t)$ of duck A_{1j} over time. We limit the speed of ducks to two cells per time step so that they cannot fly arbitrarily fast. Consequently, absolute values of the horizontal component (α) and the vertical component (β) of the velocity vector are bounded to two cells. Let $sign(\alpha)$ and $sign(\beta)$ return signs of α and β, i.e. 1 for positive and -1 for negative, respectively. Both position and velocity of birds are represented as 2D vectors; the update is therefore simply vector additions.

$$(\alpha, \beta) = v_{1j}(t) + separation(A_{1j}) + align(A_{1j}) + cohesion(A_{1j})$$

$$v_{1j}(t+1) = (sign(\alpha) min(|\alpha|, 2), sign(\beta) min(|\beta|, 2))$$

$$g_{1j}(t+1) = g_{1j}(t) + v_{1j}(t+1)$$

Separation: If duck a is close to another duck or goose b, i.e. within ϵ cells, then a flies away from b

$$separation(a) = \sum_{distance(a,b) \leq \epsilon} a.position - b.position$$

Alignment: Duck a changes its velocity by λ% towards the average velocity of its neighboring ducks

$$align(a) = \left(\frac{\sum_{duck(b)}^{neighbor(a,b)} b.velocity}{k} - a.velocity\right) \times \frac{1}{\lambda}$$

Cohesion: Duck a moves by γ% towards the center of its neighboring ducks

$$cohesion(a) = \left(\frac{\sum_{duck(b)}^{neighbor(a,b)} b.position}{k} - a.position\right) \times \frac{1}{\gamma}$$

The model for geese follows in a similar manner except that their maximum speed is three cells per time step.

3.2.2 Emergence Verification

Our proposed approach returns only a set of emergent property states. To verify that these states contain emergent properties, we show how the well-known flocking of birds emegence is derived from L_ξ. The initial system state is given as the state at time t = 0 in Figure 2. For ease of visualization, we distinguish ducks from geese using a star ($*$) symbol and bolded cell. $<j, (\alpha, \beta)>$ denotes a bird instance j, with velocity (α, β). Figure 2 shows a simulation of the system when $\epsilon = 2$, $\lambda = 10$, and $\beta = 8$. We observe that the birds keep flying in the same pattern since t = 4. Moreover, the system gets back to the system state $S(4)$ at time t = 12: $S(12) = S(4)$. Hence, $L^I_{whole} = \{S(0), S(1), \ldots, S(11)\}$.

L_{sum} is calculated using the superimpose operator as $L_{sum} = \oplus(L(A_{11}), \ldots, L(A_{25}))$. By definition and following our discussion above, L_{sum} tends to be very large, even for small problem sizes. As such, we consider for illustration two geese A_{23} and A_{25}. For simplicity, the superimposition of two birds, $L(A_{23})$ and $L(A_{25})$, is shown in Figure 3. Formally,

$$\begin{aligned}
&\oplus(L(A_{23}), L(A_{25})) \\
&= L(A_{23}) \oplus (L(A_{25})) \cup L(A_{25}) \oplus (L(A_{23})) \\
&= \{((2,(1,1),(1,1))fff\ldots(2,(1,2),(1,0))fff\ldots), \\
&\quad (fff\ldots(2,(2,2),(1,0))fff\ldots), \\
&\quad (fff\ldots(2,(8,2),(1,0))fff\ldots(2,(8,8),(1,1))\}
\end{aligned}$$

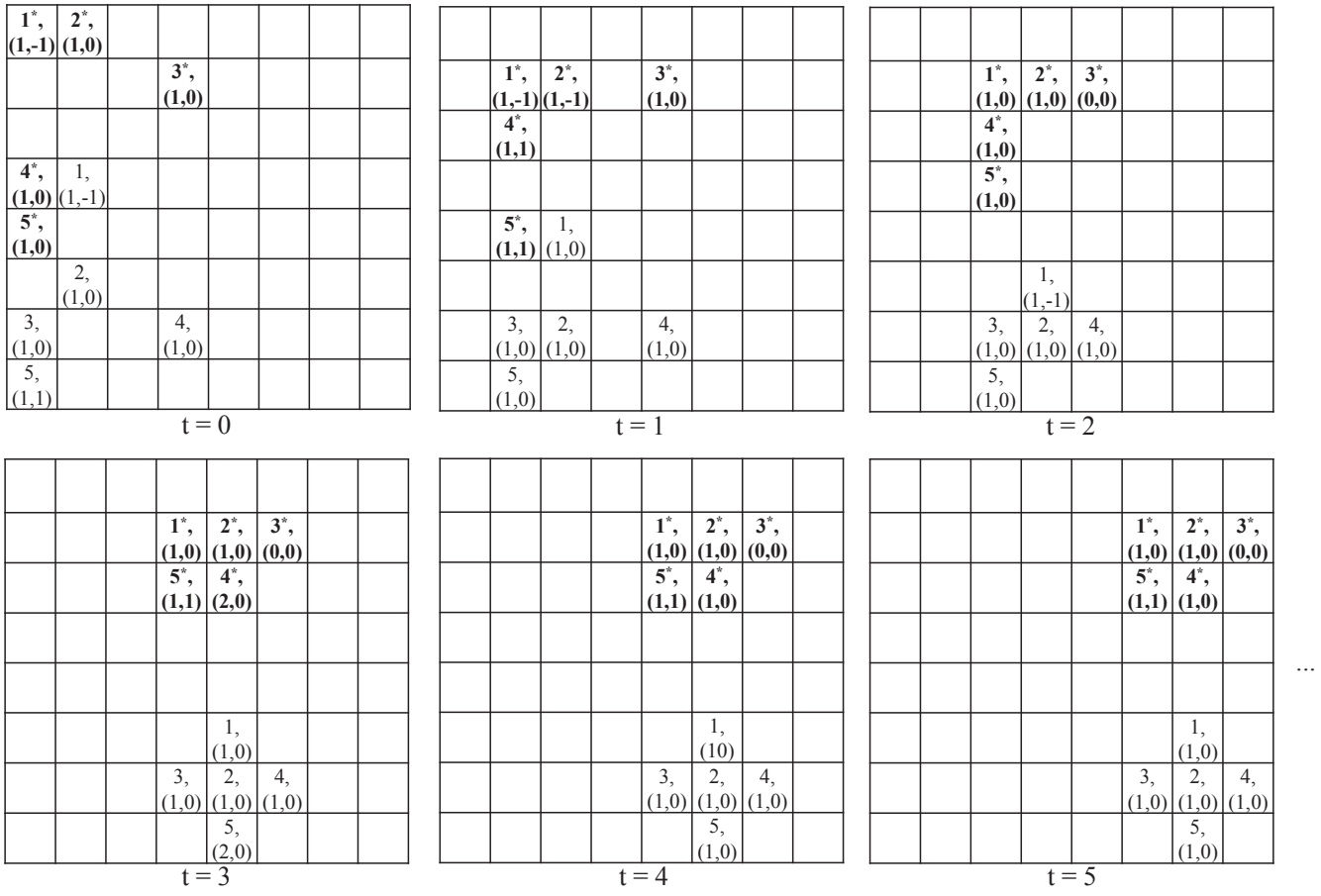

Figure 2: Snapshot of Emergent Property States

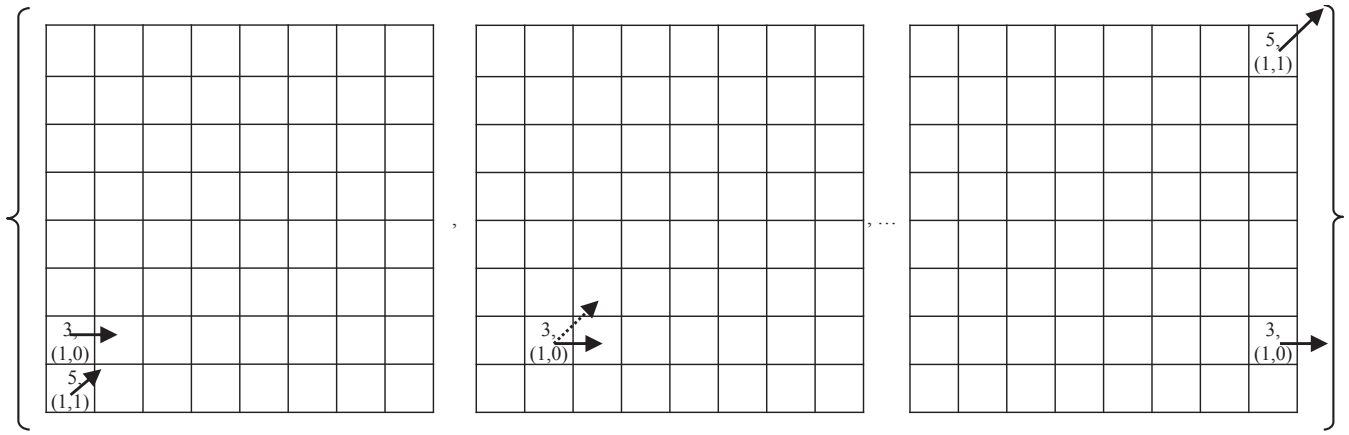

Figure 3: Example of $\mathbf{L(A_{23}) \oplus (L(A_{25}))}$

$L_(A_{23})$ and $L_(A_{25})$ are behaviors of agent A_{23} and A_{25} respectively. f represents an empty cell and a tuple $<i,(x,y),(\alpha,\beta)>$ represents the state of a bird of type i at cell (x,y) with velocity (α,β). For example, $<2,(1,2),(1,0)>$ represents a goose locating at cell $(1,2)$ with velocity $(1,0)$. Note that the x-axis is horizontal and oriented from left to right, and the y-axis is vertical and oriented from bottom to top. In the example of ten birds, L_{sum} contains one common state, S(0), with L^I_{whole}. Therefore, $L_\xi = \{S(1), S(2), \ldots, S(11)\}$. Furthermore, we assume that a group of birds form a flock if at least five birds of the same type fly *together*, with each bird has at least one immediate neighbor of the same type. Ten emergent property states from $S(2)$ to $S(11)$ therefore have flocking emergent property. In addition, flocking patterns in $S(3),\ldots,S(11)$ are identical, regardless of different positions of the birds forming the flock. In other words, we have two types of the known flocking emergent property: one type in $S(2)$ and one type in $S(3),\ldots,S(11)$. We leave the unknown emergent property state $S(1)$ for the future work.

4. EVALUATION

In this section, we present the theoretical and experimental analysis of our approach. Since we derived the sets of states for L^I_{whole}, L_{sum}, and L_ξ, we present the complexity in terms of the number of states. We implemented our approach using a Java program, and our experimental analysis quantifies the state size, by varying the number of birds.

4.1 Theoretical Analysis

The complexity of deriving L_ξ ($O(L_\xi)$) consists of two parts: complexity of L^I_{whole} ($O(L^I_{whole})$) and complexity of L_{sum} ($O(L_{sum})$) .

$$O(L_\xi) = O(L^I_{whole}) + O(L_{sum}) \tag{14}$$

Because detecting emergence is to differentiate system states that appear when taking into account interactions of agents, but not when regarding them separately, it is reasonable to use the number of system states as a complexity measure. Key complexity factors include: the environment size (2D grid of size x by y), the number of agent types (m), the number of agents (n), and the number of possible states an agent can take (s). We derive $O(L^I_{whole})$ and $O(L_{sum})$ in the *worst case*. Let $n = n' + n''$ where n' is the number of mobile agents, and n'' is the number of stationary agents.

$\boldsymbol{O(L^I_{whole})}$: Given a position, an agent can take one of s states. Moreover, stationary agents are fixed in n'' positions. There are $\binom{xy-n''}{n'}$ possibilities for allocating n' mobile agents into the remaining $xy - n''$ positions. Hence,

$$O(L^I_{whole}) = O(\binom{xy-n''}{n'} s^n) \tag{15}$$

$\boldsymbol{O(L_{sum})}$: Without considering the interactions of agents, a mobile agent, in the worst case, can arbitrarily move to any position (cell) in the environment. Hence, the upper bound complexity of superimposing individual behaviors for all agents is:

$$O(L_{sum}) = O((xy)^{n'} s^n) \tag{16}$$

For example in the game of life, agents are stationary, i.e. $n' = 0$ and $n'' = n$, and $O(L^I_{whole}) = O(L_{sum}) = O(s^n))$. If all agents are mobile, i.e. $n' = n$ and $n'' = 0$, then $O(L_{sum}) = O((xy)^n s^n)$, which is much larger than $O(L^I_{whole})$ $= O(\binom{xy}{n} s^n)$. This is because the summing operation involves all combinations of individual agents' behaviors, including those that could never happen in practice, as discussed above.

To reduce $O(L_{sum})$, we consider the fact that some results from the superimposition cannot be possible due to system wide rules, so we eliminate these from the calculation. For example, in a traffic junction model, car A following car B in a one-lane road cannot move ahead of B at any point in time. Adding system constraints that can be obtained from the system specification to the superimposition process to reduce L_{sum} to L^P_{sum} is part of our future work. The more known constraints we add, the smaller L_{sum} is.

4.2 Experimental Results

We implemented the boids model as a Java simulator to further understand the relationship among L^I_{whole}, L_{sum}, and L_ξ. We also analyzed how interactions of agents affect the size of L_ξ with respect to the size of L^I_{whole}. As L_{sum} suffers from state-space explosion, we varied the number of birds from four to ten, with equal numbers of ducks and geese. A difference between ducks and geese is maximum speed, which is two cells per time step for ducks and three cells per time step for geese. Position and velocity of birds are initialized randomly. Both ducks and geese follow three behavior rules defined in Section 3.2. Given an initial system state, at some point of time t, the system will arrive in a state that has already happened at time $t' < t$ because the number of possible system states is finite, even if it can be very large. Due to the assumption that agents' behavior rules do not change over time, the simulation loops afterwards. As a result, the size of L^I_{whole} is the number of distinct system states obtained from the beginning until time t. L_{sum}, on the other hand, is computed over all possible combinations of isolated agents' behaviors with respect to the initial system state. The initial system state belongs to both L^I_{whole} and L_{sum}.

For every experiment, we ran the simulation ten times and took the average number of states as shown in Table 3. As

number of birds	number of states			$\frac{L_\xi}{L^I_{whole}}$
	L^I_{whole}	L_{sum}	L_ξ	
4	13	767	6	0.46
6	18	70,118	12	0.67
8	13	509,103	9	0.69
10	26	13,314,066	23	0.88

Table 3: Sizes of L^I_{whole}, L_{sum}, and L_ξ

expected, the size of L_{sum} is large and increases exponentially with the number of birds. For instance, L_{sum} grows by 90 times when the number of birds changes from four to six. Another interesting observation is that L_ξ / L^I_{whole} tends to increase with the number of birds. In other words, more interactions (interdependences) among birds lead to more emergent property states that cannot be derived by summing the independent agents' behaviors. Furthermore, the number of common elements between L^I_{whole} and L_{sum} is small. Consequently, computation is wasted on calculating L^{NP}_{sum} states that are not feasible in practice. However, when the impact of neighboring agents on an agent is not strong, then L^{NP}_{sum} tends to be small. For example, agents

do not interact frequently because they are far from each other, such as when the number of agents is much smaller than the number of cells in the environment.

5. CONCLUSION

In this paper, we formalized emergence using an extended cooperating array grammar system so as to verify the existence and the size of emergent property states in multi-agent systems. In weak emergence, the set of emergent property states for a given system (L_ξ) is derived by taking the difference between the set of observed system states due to agent interactions (L_{whole}), and the set of system states obtained by combining the behaviors of individual agents (L_{sum}). However, this broad definition suffers from state explosion. As a first contribution, we introduced a tighter definition of weak emergence to reduce the size of the system state space. In studying a system, users focus only on modeling agent interactions of interest ($L^I_{whole} < L_{whole}$). Though all combinations of L_{sum} are mathematically possible, a large number of these system states may not be feasible because of system constraints. Thus, L_{sum} can be reduced to L^P_{sum}, i.e. set of possible combinations of agents' behaviors.

In the second contribution, we proposed a grammar-based formalism of emergence that models agents of different types, mobile and static agents, as well as open systems with agents arriving and departing over time. Theoretical analysis reveals a number of observations, e.g. the complexity of L_{sum} increases exponentially with the number of agents. These observations were also verified experimentally using a boids model with two types of birds. In addition, we showed how a known emergent property such as bird flocking can be extracted from the emergent property states. In terms of system state-space size, our approach is comparable with model checking. State-of-the-art model checkers can address state space of about 10^9 states with explicit state-space enumeration, and over 10^{20} with techniques for alleviating state explosion [3]. Preliminary experimental results show that our simulator can handle L_{sum} of about 10^8 states. L_{sum} can be further reduced by identifying system-specific constraints that eliminate impossible states when summing individual agents' behaviors.

This paper provides a first step towards advancing our understanding of emergence but much work remains to be done. New techniques are needed to extract *known* emergent properties, and to identify new (or *unknown*) emergent properties from the set of emergent property states. For example, if emergent properties are known, our formalism facilitates post-mortem emergence analysis to determine the causes of emergence. As our approach is both compute and memory intensive, more efficient state representations, such as bit state hashing and state vector compression used in model checking, are required. We are exploring the application of this approach to study emergence in concurrent program verification. Given a program and its specification, the cause-and-effect of both known and unknown properties, such as deadlock, can be analyzed by examining the corresponding system states. This approach compliments model checking where the main goal is to verify predetermined behavioral properties of a given system.

6. ACKNOWLEDGMENTS

This work is supported by the National University of Singapore under grant number R-252-000-470-112.

7. REFERENCES

[1] R. Abbott, Emergence Explained: Abstractions Getting Epiphenomena to Do Real Work, Complexity, 12(1):13-26, 2006.

[2] N. A. Baas and C. Emmeche, On Emergence and Explanation, Intellectica, pages 67-83, 1997.

[3] C. Baier and J. Katoen, Principles of Model Checking, The MIT Press, 2008.

[4] Y. Bar-Yam, A Mathematical Theory of Strong Emergence Using Multiscale Variety, Complexity, 9(6):15-24, 2004.

[5] M. A. Bedau, Weak Emergence, Philosophical Perspectives: Mind, Causation, and World, 1997.

[6] M. A. Bedau, Downward Causation and the Autonomy of Weak Emergence, Principia 3, 3:5-50, 2003.

[7] E. Bonabeau and J. Dessalles, Detection and Emergence, Intellectica, 25(2), 1997.

[8] D. J. Chalmers, Strong and Weak Emergence, The Re-emergence of Emergence, Oxford University Press, 2006.

[9] W. K. V. Chan, Interaction Metric of Emergent Behaviors in Agent-based Simulation, Proc of Winter Simulation Conference, pages 357-368, 2011.

[10] C. C. Chen, Complex Event Types for Agent-based Simulation, PhD thesis, University College London, 2009.

[11] J. P. Crutchfield, Is Anything Ever New? Considering Emergence, Complexity: Metaphors, Models, and Reality, G. Cowan, D. Pines, and D. Melzner (eds.), SFI Series in the Sciences of Complexity XIX:479-497. Addison-Wesley, 1994.

[12] V. Darley, Emergent Phenomena and Complexity, Artificial Life IV, pages 411-416, 1994.

[13] J. Deguet, L. Magnin, and Y. Demazeau, Elements about the Emergence Issue: A Survey of Emergence Definitions, ComPlexUs, 3:24-31, 2006.

[14] G. B. Dyson, Darwin Among the Machines: The Evolution of Global Intelligence, Perseus Books Group, 1998.

[15] D. Fisch, M. Janicke, B. Sick, and C. Muller-Schloer, Quantitative Emergence - A Refined Approach Based on Divergence Measures, Proc of 4th IEEE International Conference on Self-adaptive and Self-organizing Systems, pages 94-103, 2010.

[16] J. Fromm, Types and Forms of Emergence, Complexity Digest, vol. 25(3), 2005.

[17] R. Gore and P. F. Reynolds, Applying Causal Inference to Understand Emergent Behavior, Proc of Winter Simulation Conference, pages 712-721, 2008.

[18] P. Haglich, C. Rou, and L. Pullum, Detecting Emergence in Social Networks, Proc of 2nd IEEE International Conference on Social Computing, pages 693-696, Minneapolis, MN, USA, 2010.

[19] B. Heath, R. Hill, and F. Ciarallo, A Survey of Agent-based Modeling Practices (January 1998 to July 2008), Journal of Artificial Societies and Social Simulation, 12(4):9, 2009.

[20] J. H. Holland, Emergence: From Chaos to Order, Addison Wesley, 1997.

[21] C. W. Johnson, What are Emergent Properties and How Do They Affect the Engineering of Complex Systems? Reliability Engineering and System Safety, 12:1475-1481, 2006.

[22] A. Kubik, Toward a Formalization of Emergence, Artificial Life IX, 9(1):41-65, 2003.

[23] Z. Li, C. H. Sim, and M. Y. H. Low, A Survey of Emergent Behavior and Its Impacts in Agent-based Systems. Proc of IEEE International Conference on Industrial Informatics, pages 1295-1300, 2006.

[24] E. J. Manley and T. Cheng, Understanding Road Congestion as an Emergent Property of Traffic Networks, Proc of 14th World Multi-conference on Systemics, Cybernetics and Informatics, pages 25-34, 2010.

[25] T. Moncion, P. Amar, and G. Hutzler, Automatic Characterization of Emergent Phenomena in Complex Systems, Journal of Biological Physics and Chemistry, 10:16-23, 2010.

[26] M. Mnif and C. Muller-Schloer, Quantitative Emergence, Proc of IEEE Mountain Workshop on Adaptive and Learning Systems, pages 78-84, Logan, UT, USA, 2006.

[27] J. C. Mogul, Emergent (Mis)Behavior vs. Complex Software Systems, Proc of 1st ACM SIGOPS/EuroSys European Conference on Computer Systems, volume 40, pages 293-304, New York, USA, 2006.

[28] M. E. J. Newman, Complex Systems: A Survey, American Journal of Physics, 79:800-810, 2011.

[29] I. Norros, B. J. Prabhu, and H. Reittu, Flash Crowd in a File Sharing System Based on Random Encounters, Proc of Workshop on Interdisciplinary Systems Approach in Performance Evaluation and Design of Computer & Communications Systems, New York, USA, 2006.

[30] T. O'Conner, Emergent Properties, American Philosophical Quarterly, 31(2):91-104, 1994.

[31] D. T. Pele and A. M. Tepus, Information Entropy and Efficient Market Hypothesis, Proc International Conference on Applied Economics, 2011.

[32] M. Randles, H. Zhu, and A. Taleb-Bendiab, A Formal Approach to the Engineering of Emergence and its Recurrence, Proc of 2nd International Workshop on Engineering Emergence in Decentralized Autonomic Systems, 2007.

[33] C. W. Reynolds. Flocks, Herds and Schools: A Distributed Behavioral Model, Proc of 14th Annual Conference on Computer Graphics and Interactive Techniques, pages 25-34, New York, NY, USA, 1987.

[34] C. Rou, A. Vanderbilt, M. Hinchey, W. Truszkowski, and J. Rash, Properties of a Formal Method for Prediction of Emergent Behaviors in Swarm-based Systems, Proc of 2nd IEEE International Conference on Software Engineering and Formal Methods, September 2004.

[35] A. K. Seth, Measuring Emergence via Nonlinear Granger Causality, Artificial Life XI, pages 545-552, 2008.

[36] C. E. Shannon, A Mathematical Theory of Communication, Bell System Technical Journal, 27(3):379-423, 2000.

[37] C. Szabo and Y. M. Teo, An Integrated Approach for the Validation of Emergence in Component-based Simulation Models, Proc of Winter Simulation Conference, pages 242:1-242:12, 2012.

[38] C. Szabo and Y. M. Teo, An Objective-based Approach for Semantic Validation of Emergence in Component-based Simulation Models, Proc of 26th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation, pages 155-162, ZhangJiaJie, China, July 15-19, 2012.

[39] Y. M. Teo and C. Szabo, Semantic Validation of Emergent Properties in Component-based Simulation Models, Book Chapter in Ontology, Epistemology, and Teleology of Modeling and Simulation - Philosophical Foundations for Intelligent M&S Applications, edited by Andreas Tolk, pages 319-333, Springer-Verlag, 2013.

[40] B. Zhan, D. N. Monekosso, P. Remagnino, S. A. Velastin, and L. Q. Xu, Crowd Analysis: A Survey. Machine Vision and Application, 19(5-6):345-357, September 2008.