Proceedings of the 2012 Winter Simulation Conference
C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, eds.

AN INTEGRATED APPROACH FOR THE VALIDATION OF EMERGENCE IN COMPONENT-BASED SIMULATION MODELS

Claudia Szabo

Department of Computer Science
The University of Adelaide
Adelaide, AUSTRALIA

Yong Meng Teo

Department of Computer Science
National University of Singapore
SINGAPORE

ABSTRACT

Emergent properties are becoming increasingly important as systems grow in size and complexity. Despite recent research interest in understanding emergent behavior, practical approaches remain a key challenge. This paper proposes an integrated approach for the identification of emergence with two perspectives. A post-mortem emergence analysis requires a-priori knowledge about emergence and can identify the causes of emergent behavior. In contrast, a live analysis, in which emergence is identified as it happens, does not require prior knowledge and relies on a more rigorous definition of individual model components in terms of *what* they achieve, rather than *how*. Our proposed approach integrates reconstructability analysis in the validation of emergence included in our proposed component-based model development life-cycle.

1 INTRODUCTION

The analysis of individual, interacting components in a complex system cannot fully define the behavioral properties of the system (Johnson 2006, Mogul 2006). These properties, called emergent properties, are increasingly becoming important as systems grow in complexity, coupling, and geographic distribution (Bedau 1997, Holland 1999, Johnson 2006, Mogul 2006). Examples of emergent properties include the flocking of birds (Reynolds 1987), connection patterns in data extracted from social networks (Chi, Chan, Seow, and Tam 2009), trends in big data analytics (Fayyad and Uthurusamy 2002), the Ethernet capture effect (Ramakrishnan and Yang 1994), load-balancer failures in multi-tiered distributed systems (Mogul 2006), and router synchronization problems (Floyd and Jacobson 1993) among others. While the previous examples have limited negative effects, malign emergent properties have been identified in power grid blackouts occurring despite the functioning within bounds of all equipment (Force 2004). Another example of negative emergence occurred in the UK health system in 2009, where a new requirement not to keep patients waiting resulted in unwanted consequences. These included patients being given unnecessary treatment to move them down the waiting list and the invention of a “hello nurse” to simply greet patients upon arrival to reduce waiting time (Chen, Nagl, and Clack 2007). In this paper, we focus on component-based simulation as a mean of studying complex systems, where individuals in the complex system are abstracted as model components, and the system is studied as a simulation.

Because of undesired and unpredictable effects that lead to less credible systems that are difficult to manage, techniques for the identification of emergent properties are becoming of crucial importance (Chen, Nagl, and Clack 2007, Gore and Reynolds 2007). It is also important to determine whether identified emergent behavior is detrimental or harmless to the system, in other words to perform emergence *validation*, which, to the best of our knowledge, has not been addressed by current work. Most approaches focus only on the observation of emergence in various biological, social, and AI contexts, on its philosophical classification (Holland 1999), and less on measuring and advancing our understanding in the cause-and-effect of emergence (Szabo and Teo 2012b). Previous studies show that despite a plethora of emergence examples that have been identified and classified (Chen, Nagl, and Clack 2009, Holland 1999, Kubik 2003, Mogul 2006), few instances have been measured and explained (Szabo and Teo 2012b). A difficulty lies in defining the variables that describe the system sub-components, or the *micro-level*, and the system as a whole, or the *macro-level*, and the relationships between the micro-level and macro-level. There is a critical need to understand the interaction between components at the micro-level the composed models, and how this interaction leads to emergence at the macro-level. Moreover, existing approaches (Chen, Nagl, and Clack 2007, Kubik 2003, Seth 2008) are currently only applied to simple models such as flocks of birds, and have limiting assumptions and constraints when applied to more complex systems (Szabo and Teo 2012b). A key challenge is the need for abstractions of the micro and macro levels. These abstractions are difficult to achieve in an automated manner, and hence most approaches rely on a post-mortem observation of the simulation by a system expert (Gore and Reynolds 2008), instead of a live emergence identification without a-priori knowledge (Kubik 2003). Moreover, current approaches assume that the system state, sub-components and emergent behavior can be captured using a single variable and thus are not applicable to cases where more than one variable is needed to capture micro or macro levels.

There is a gap between the abstraction of the macro (the system) and micro (the system sub-components) levels in simple models and more complex models, and between having prior knowledge of emergence, and having no information at all. It would be beneficial if these orthogonal perspectives were united in a single approach to further the understanding of emergent behavior in complex systems as they are developed and deployed in real life (Chen, Nagl, and Clack 2007, Holland 1999, Seth 2008). In this paper, we propose an integrated approach that permits the identification of emergent behaviors in component-based simulation models from two perspectives. If knowledge about emergence is available beforehand, our approach will identify the model components and their interactions that are the most likely causes of emergent behavior. Otherwise, our live emergence analysis approach aims to identify and validate emergence as it happens. The contributions of this paper include:

- An integrated approach for detecting emergent behavior. Firstly, a post-mortem analysis identifies the causes, in terms of model component states, that led to a previously defined emergent property. Secondly, our live emergence identification approach relies on our proposed model component definition to identify emergent behavior as it happens.
- A novel definition of model components in terms of *what* they achieve rather than *how* proposes to offer a live analysis of emergence, as it happens in the simulation. This definition will also permit the visualization of model component interaction and states leading to emergence and further the understanding of the studied system.

This paper is organized as follows. We compare and contrast related work in Section 2. Section 3 presents our proposed approach, discussing post-mortem and live emergence perspectives. An example is analyzed in Section 4. We present our concluding remarks and discussion of further work in Section 5.

2 RELATED WORK

Emergent behavior in component-based simulation model development has been highlighted since the 1990's by Page and Opper (Page and Opper 1999), which propose a formal framework for analyzing the complexity of composition and emergence. Their proposed definition uses the given components a and b that are composed as $a \diamond b$, an objective o, and the "satisfies" operator $\vDash$. If $a \vDash o$, then a satisfies objective o. If $a \nvDash o$ and $b \nvDash o$, but $(a \diamond b) \vDash o$, then we can say that the composition is *emergent*.

Gore and Reynolds denote emergence as a specific variable value and propose to highlight the lines in the simulation source code that cause that particular value (Gore and Reynolds 2008). They further propose a taxonomy for analyzing emergent behavior based on reproducibility, predictability, and temporality (Gore and Reynolds 2007). *Reproducibility* refers to the repeatability of a simulation for a given set of inputs. *Predictable behaviors* enable selective sampling towards testing user hypotheses. *Temporality* distinguishes between the simulation reaching a final state and residing in a particular state. The proposed taxonomy allows system experts to classify a particular emergent behavior. Later work uses an emergent property definition of a specific variable and its value in the simulation source code and identifies the lines of code that led to that value.

Another approach to identify emergence proposes to look at a measure of the interaction between agents in an agent-based model (Chan 2011). The interaction metric is an agent-specific counter that increases as the agent interacts with other agents in the environment. Emergence is said to appear if the interaction measure deviates from normality. This approach provides a straightforward measure of emergence. However, the study considers only simple models such as Conway's Game of Life (Gardner 1970) and the flock of birds model (Reynolds 1987) in which emergence is a direct result of agent interaction, but does not address cases where emergence is a result of indirect interaction between agents.

In the complex systems domain, emergence validation approaches can be classified in three main categories, namely, *grammar-based*, *variable-based*, and *event-based*. *Grammar-based methods* employ two grammars, L_{WHOLE} to describe the properties of the system as a whole, and L_{PARTS} to describe the properties obtained from the reunion of the parts (Kubik 2003). Emergence is defined as the difference between L_{WHOLE} and L_{PARTS} (Kubik 2003). While L_{PARTS} is calculated from the behavior of agents without considering the agent interaction with the environment, L_{WHOLE} is obtained as a reunion of all the symbols generated by agents. This method does not require prior observation of the system to identify possible emergent properties or behaviors, which makes it suitable for large composed models where such observations are almost impossible. However, the calculation of L_{PARTS} considers all possible combinations of agents states and thus is difficult to scale.

In *variable-based methods*, a specific variable is chosen to describe emergence. Changes in the values of this variable are said to signify the presence of emergent behavior (Seth 2008). Seth (Seth 2008) proposes G-emergence, a measure based on Granger causality to establish the relationships between a macro-variable, representing a system property, and micro-variables, representing properties of the system sub-components. This approach provides a clear process to identify emergence because it looks at measurable quantities found in the system state, which is defined as the reunion of all sub-systems states. However, finding a good variable to describe a system can be a difficult task that requires system expert intervention and extensive observations of the system. Moreover, while Granger causality can be adapted to handle more than one-to-one dependencies, G-emergence has yet to be defined in this case, and might not apply when the macro-variable depends on more than one micro-variable.

In *event-based methods*, behavior is defined as a series of events which change a system or a sub-system state (Chen, Nagl, and Clack 2007). A *simple event* type signifies a change in a sub-system state. It is associated with a transition and has a duration. A *complex event* is defined as being either a simple event or constituted from two complex events linked by a relationship. Emergence, defined as a complex event, can then be reduced to a sequence of simple events. However, a detailed definition of the emergent property in terms of simple and complex events is required for the analysis.

3 PROPOSED APPROACH

If emergent behavior is known beforehand, a key challenge is to identify the causes of emergence using a method that is independent of the number of model components, and with an acceptable level of abstraction to reflect model behavior and attributes. In the absence of an identified emergent behavior, the challenge is the representation of the micro and macro levels to identify unexpected behavior as it appears.

We propose an integrated approach that captures these orthogonal perspectives, as shown in Figure 1. Firstly, if emergent behaviors are identified beforehand for a particular composed model, their cause can be analyzed using a definition of emergent properties and an emergence taxonomy that leverages on previous classification work (Holland 1999). In this *post-mortem emergence analysis*, we propose to identify the causes, in terms of model component attributes and values, that lead to emergence. In contrast with current work (Seth 2008), our proposed approach is not limited by the number of model components in the composed model and can capture model characteristics with a high level of detail. The emergent behaviors that have been identified beforehand are also saved into an emergent property repository. We propose to structure this repository using characteristics from various emergence taxonomies that look at a property's repeatability, reproducibility, and temporality, as well as whether it is a strong or weak emergence (Bedau 1997, Gore and Reynolds 2007). These characteristics will help further the understanding of unknown behavioral properties as they appear.

If an emergent behavior is not known beforehand, we propose a novel objective-based representation of model components that permits the identification of emergence as it appears in the simulation. Our *live emergence analysis* defines a meta-component that specifies model components in terms of *what* they achieve rather than *how*. We propose to obtain the composed model state from the states of its model components. This constructed state is compared with the observed simulation state and significant differences are noted, also based on an ontology-based representation of specific domain knowledge. We propose the use of COSMO, our proposed ontology for component-based simulation, to represent simulation and domain knowledge (Teo and Szabo 2008). Subsequently, properties identified as emergent are saved into the emergent property repository. We envisage this repository as a collection of properly specified and defined emergence properties that leverage on existing work in the classification of emergence with respect to type, application domain, and specific occurrence among others. We employ reconstructability analysis (Cavallo and Klir 1979) in two directions, namely, to determine the model components whose interaction leads to emergence (in post-mortem), and to calculate a composed model state from the states of its model components (in live analysis). Reconstructability analysis (RA) is an approach to discrete multi-variate modeling developed in the systems community. RA decomposes the macro level into micro level relationships that are specified in terms of relations and distributions involving subsets of variables. A result of reconstructability analysis on a set of micro and macro variables specifies the inter-dependencies among the micro variables that result in a macro variable with minimum errors and reduced complexity. A variant of reconstructability analysis, called *reconstruction*, proposes the reverse of this process, namely, constructing the macro level from the micro level. Another important point in our proposed approach is the visualization and validation of emergent behavior. We propose to determine whether a particular emergent behavior is beneficial or detrimental to the system by calculating and visualizing differences between the system states when the emergent behavior occurs as a first step towards this, we propose to highlight to system experts emergent behavior that differs, with respect to system states, than expected behavior.

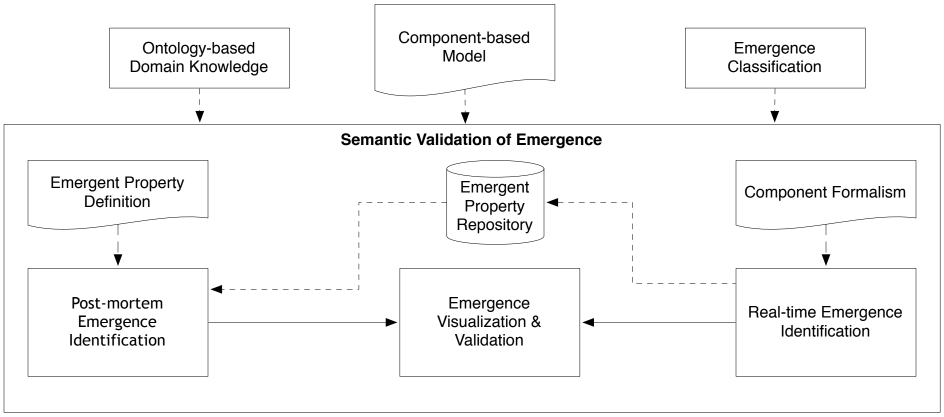

Ontology-based Domain Knowledge
Component-based Model
Emergence Classification
Semantic Validation of Emergence
Emergent Property Definition
Emergent Property Repository
Component Formalism
Post-mortem Emergence Identification
Emergence Visualization & Validation
Real-time Emergence Identification

Figure 1: Integrating Orthogonal Perspectives in Emergence Validation.

3.1 Post-mortem Emergence Analysis

Our proposed post-mortem emergence identification approach assumes a-priori knowledge of emergent properties and identifies the causes that lead to their appearance using reconstructability analysis (RA). Our approach relies on a representation of model components that captures component attributes and behavior. More formally, model components are represented by *meta-components*, $C_i = \langle R, A_i, B_i \rangle$, which describe the component *required attributes* R, *specific attributes*, A, and *behavior*, B, and are extensively used throughout the life-cycle of a composed model. The required attributes are common to all components and are generally employed for version control, e.g., `author, location, lastUsed`. Examples of specific attributes include `interArrivalTime`, `numJobsServiced`, or `speed`, `direction` etc. The component behavior describes the data that it receives and outputs as a set of states. The transitions between states are defined as a set of triggers expressed in terms of input, time and conditions.

Our approach has three main steps as shown in Figure 2. Firstly, an emergent property is observed and specified as EP. Secondly, to identify the causes of EP, the composed model simulation is executed and the model components, representing the *micro* level, are observed throughout the simulation run and recorded as μP. The values of EP, or the *macro* level, are also recorded. Thirdly, we employ RA to determine the relationship between μP and EP.

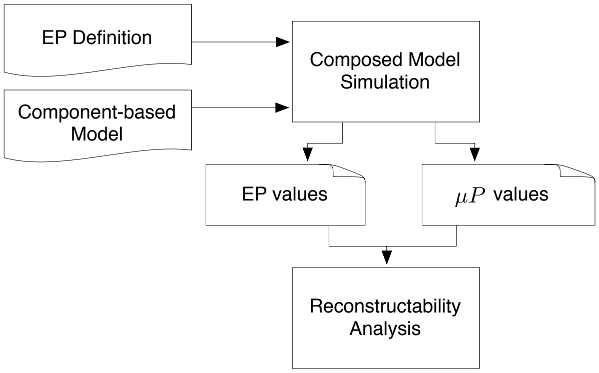


Figure 2: Post-mortem Emergence Analysis.

For a set of recorded values for the micro properties μP_{ij} at every simulation time step j, and the set of recorded macro properties EP_j, we construct a system in which the inputs are μP_i and the output is EP, and the observations are μP_{ij} and EP_j at every simulation step j. Reconstructability analysis decomposes the system (μP_{ij} and EP_j) into various subparts (that include subsets of μP_{ij}, and EP_j) that are then assessed for statistical significance. This allows us to identify the interaction of model components in μP_i that has the highest influence on the emergent behavior EP. For example, for inputs A, B, and output C, five model structures will be considered, namely, ABC, AB:AC:BC, AB:AC, AB:BC, AB:C. In model ABC, A and B interact in their joint effect on C. Structure AB:C signifies that the output is independent of the input. A structure AB:AC with the highest statistical significance means that the model has one predicting component, AC. Reconstructability analysis promises to scale well in terms of the number of properties and their observations. Moreover, in contrast to existing work, it also permits the formal and statistical analysis of the influence of component interaction on the emerging behavior, which increases the insight into the system execution.

3.2 Live Emergence Analysis

When previous knowledge about emergent properties is not available, we propose a two-step approach that consists of the *identification* of an *emergence set*, and *validation* of the composed model states, as shown in Figure 3. Existing specifications of model components that formalize the model component behavior do not capture what the model component achieves, but focus on how the model component behaves. A description of the purpose or objective of a model component that could be used to deduce the behavior of the composed model will facilitate the identification of emergent behavior as it allows for the identification of semantically different model states.

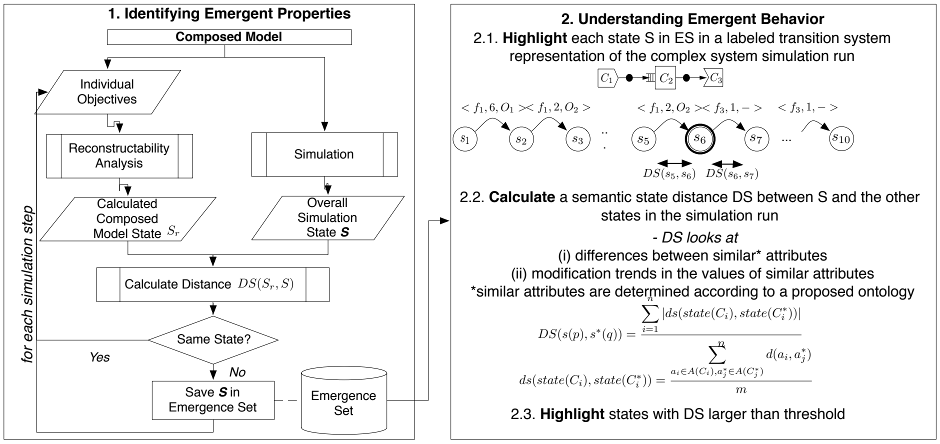

1. Identifying Emergent Properties
Composed Model
Individual Objectives
Reconstructability Analysis
Simulation
Calculated Composed Model State S_r
Overall Simulation State *S*
Calculate Distance $DS(S_r, S)$
Same State?
Yes
No
Save *S* in Emergence Set
Emergence Set
for each simulation step
2. Understanding Emergent Behavior
2.1. Highlight each state S in ES in a labeled transition system representation of the complex system simulation run
C_1
C_2
C_3
$< f_1, 6, O_1 >< f_1, 2, O_2 >$
$< f_1, 2, O_2 >< f_3, 1, - >$
$< f_3, 1, - >$
s_1
s_2
s_3
s_5
s_6
s_7
s_{10}
$DS(s_5, s_6)$
$DS(s_6, s_7)$
2.2. Calculate a semantic state distance DS between S and the other states in the simulation run
- DS looks at
(i) differences between similar* attributes
(ii) modification trends in the values of similar attributes
*similar attributes are determined according to a proposed ontology
$DS(s(p), s^*(q)) = \frac{\sum_{i=1}^{n} |ds(state(C_i), state(C_i^*))|}{n}$
$ds(state(C_i), state(C_i^*)) = \frac{\sum_{a_i \in A(C_i), a_j^* \in A(C_j^*)} d(a_i, a_j^*)}{m}$
2.3. Highlight states with DS larger than threshold

Figure 3: Objective-based Approach for Identification of Emergent Properties.

Towards this, we enhance each component definition C_i with an added objective $o(C_i)$ that describes *what* the component achieves. The objective is defined using both textual description and a set of variables, $var(o(C_i))$, with a type and a range of values that signify the objective is met. More formally,

$$var = \{(type, values) | type \in COSMO\}$$

where the variable type is defined in COSMO, and can be of primitive types such as `int` or more complex types. The variables also represent a subset of the attributes defined in the state machine. Thus,

$$o(C_i) = \{(description, var) | var \in A_m\}$$

As such, the variables defining the component objective can change value with transitions. When these variables reach values in the specified range, the objective is said to be met. For the *Source* component C_1 in Figure 3, the objective can be defined as

$$o(C_1) = (description, var(o(C_1)))$$

where

$$\begin{aligned} description = \textit{Generates jobs every 3.5 seconds on average}; var(o(C_1)) \\ = \{numJobsGenerated, interArrivalTime\}. \end{aligned}$$

The attribute *numJobsGenerated* captures the fact that the *Source* component generates jobs, whereas the *interArrivalTime* captures the second part of the objective description.

This objective-based definition of a component captures the relevant attributes that define the component behavior, reducing the number of attributes considered in live analysis. However, several assumptions are in place. Firstly, the objective must be defined as specified above, by the model component creator. Secondly, the variables defining the objective must be already part of the component definition, as specified by $var \in A_m$. A higher level representation should have these objectives defined as independent of the component attributes, with additional rules specifying when these objectives are met. This approach also has its disadvantages and its study is part of our future work.

3.2.1 Identifying Emergent Properties

Our live emergence analysis proposes to present to the user an emergence set that contains simulation states that are significantly different from a set of calculated composed model states. Informally, an emergent state is a simulation state that is different by a semantic distance from a calculated state. Our approach relies on the objective-based definition discussed above and presented in detail in (Szabo and Teo 2012a).

As shown in Figure 3, our analysis is executed through the simulation run. At each time moment k, our approach has three main steps. In the first step, the composed model state S_{rk} is calculated using reconstructability analysis as detailed below. In the second step, the simulation state S is calculated as the reunion of model component attributes and their values. In the third step, the semantic distance $DS(S_{rk}, S)$ is calculated. If DS has a value greater than a pre-defined threshold ε, then the simulation state S is added to an emergence set ES.

Step 1. Calculation of S_{rk}
To calculate S_{rk}, we define each model component state $s(C_i)|_o$, as the reunion of all variables *var* that are defined in its objective $o(C_i)$. The model component state $s(C_i)|_o$ is restricted to include only the model component objectives. In contrast, the simulation state S is defined as the reunion of model component states that contain *all* model component attributes and their values (including objectives). For the calculation of S_{rk} from $s(C_i)|_o$, we propose to use a simplified version of a greedy algorithm adapted from (Jones 1985).

Our algorithm incrementally adds states from the set $\{s(C_i)|_o | i = 1 \ldots n\}$ to S_{rk} that is initially empty. An added state β is one which is relevant by the measure $\gamma(\beta, S_{rk})$, where γ is informally calculated as how statistically different is S_{rk} from $S_{rk} \cup \{\beta\}$, and from previous sets $S_{rj}, j < k$, calculated at previous simulation steps. Currently, we propose a simple measure of this statistical difference, by looking at the Mann-Whitney-Wilcoxon significance tests over the sets of objective variable values from S_{rk}, $S_{rk} \cup \{\beta\}$, and the sets $S_{rj}, j < k$. In the future, we plan to study probabilistic models as discussed in (Willet and Zwick 2004).

Step 2. Observation of S
The composed model state is observed at time k is observed and collated as the reunion of all model component attributes and their values, as in Step 2 of our proposed post-mortem analysis.

Step 3. Live Emergence Identification
To determine how different S_{rk} is from S, we employ a semantic state distance metric, DS that looks at the difference between similar attributes as defined in the COSMO ontology (Teo and Szabo 2008). The semantic state distance, DS, measures the semantic differences between component attribute values for attributes a_i and a_j, as

$$d(a_i, a_j) = \begin{cases} 0 & \text{if related}(a_i,\ a_j) \text{ and } ||value(a_i) - value(a_j)|| \leq \varepsilon \\ 0.5 & \text{if related}(a_i,\ a_j) \text{ and} ||value(a_i) - value(a_j)|| > \varepsilon \\ 1 & \text{if } \nexists a_j \in A(C_i) \text{ s.t. related}(a_i,\ a_j) = \text{true} \end{cases}$$

where $related(a_i, a_j)$ signifies that a_i and a_j are related in the COSMO ontology.

If there is an unacceptable deviation in the observed parameters, i.e., $||value(a_i) - value(a_j)|| \geq \varepsilon_1$ and $DS(S_{rk}, S) \geq \varepsilon_2$, we highlight this state as a possible emergence state and add it to an *emergence set*, ES as $ES \cup \{S\}$. We repeat this for the entire simulation run.

3.2.2 Understanding Emergence

The live emergent property identification constructs an emergence set ES that contains possible candidates for emergence. Of greater interest for system experts is to determine if the identified emergent properties are beneficial or detrimental to the system. As a first step towards this validation of emergence, we propose to highlight to the user the states in ES as they appear in the simulation run. Towards this, we propose to represent the simulation run as a Labelled Transition System (LTS) (Srba 2001) and to highlight in the LTS representation the simulation states that appear in ES. Because the size of ES may be too large to facilitate proper visualization, we propose to highlight only significant ES states.

We base our proposed visualization on our previous time-based formalism for the representation of a model component (Szabo, Teo, and See 2009). The formalism relies on the definition of each model component as a mathematical function, f, as $f_i : X_i \to Y_i$, where $X_i = I_i \times S_i \times T_i$, and $Y_i = O_i \times S_i \times T_i$. I_i and

O_i are the set of input/output messages, S_i is the set of states and T_i is the set of simulation time intervals at which the component changes state. Each node in the LTS represents an annotated composition state given by the tuple $S_{j=1,m} = [\{state(C_i)_{i=\overline{1,n}}\}, f_{in},\ f_{out}]$, where $state(C_i)$ is the state of component C_i, n is the number of components, m is the number of simulation states, f_{in} is the component that generated this transition, and f_{out} is the component that executed to exit this node. Labels are the tuple <*function_name, duration, output*>, where *duration* represents the function execution time.

We identify s as the LTS states that are also in ES, and determine all their neighboring states s_{near}, i.e., the states that have either an outgoing edge towards s, or an incoming edge from s. For these states, we calculate the semantic state distance $DS(s, s_{near})$, as defined above and highlight the states s for which DS has a value higher than a threshold.

4 EXAMPLE

For simplicity, we employ a model of a flock of birds in which each component abstracts a moving bird that changes its position based on a set of simple rules that defines its current position and the position of the other birds in the flock. These rules are (i) *separation* - individual birds steer to avoid crowding the other birds in the flock (ii) *alignment* - individual steers towards the average herding of local flockmates and (iii) *cohesion* - individual moves towards the average position of local flockmates. The boid model has been shown to exhibit emergent behavior of flocking, and of flocking after encountering an obstacle, when the flock splits and reunites.

4.1 Post-Mortem Emergence Analysis

In post-mortem emergence analysis, a bird is abstracted as a model component with `position` and `speed` as attributes, among others. Birds exhibit emergent properties, such as flocking. The macro level emergent property identified a-priori as representing emergence is the fact that the relationship between the trajectory of the center of mass (CM) of the flock of birds and the trajectories of individuals in the flock corresponds to flocking behavior that is observed visually (Seth 2008). The simulation of the flock of birds is executed, and data is collected, pre-processed, and collated into a file that is then supplied to Occam3, a widely used reconstructability analysis tool (M. Zwick 2012).

Our example employs ten birds that go through a sequence of 20 position changes according to the rules specified above, starting from random positions on the drawing panel. At each simulation step, each bird changes position according to the rules specified above; we collect each bird position, as well as that of the center of mass of the flock. We reduce the number of variables by computing the distance d_i from the center of the environment, i.e., the center of the drawing panel. The values of d_i are saved into an output file, similar to that shown in Table 1 for the first ten steps.

Table 1: Distance from the Center of Drawing Panel.

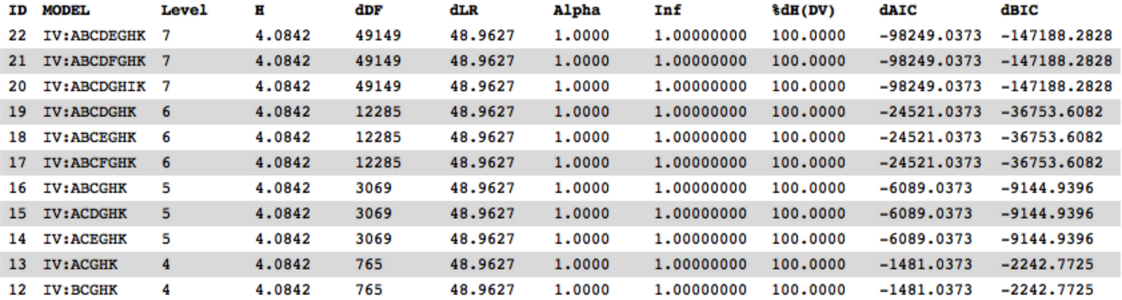
Bird 1	Bird 2	Bird 3	Bird 4	Bird 5	Bird 6	Bird 7	Bird 8	Bird 9	Bird 10	CM
53	209	128	67	43	104	121	130	118	95	69
45	174	112	65	44	91	104	111	103	81	65
40	129	88	60	43	72	81	84	80	61	61
39	76	59	51	38	49	52	53	55	38	62
38	27	42	34	37	31	19	30	31	23	69
54	32	48	11	57	9	31	53	42	36	80
59	71	69	13	75	18	70	102	81	69	101
56	113	100	42	94	51	111	153	125	109	131
58	154	136	79	118	92	155	202	171	153	169
82	195	175	125	151	140	200	247	213	202	213

To further reduce the number and values of the variables, we further discretize the distance values, as $floor(d_i/100)+1$. The results are then transformed into an Occam3 input file, as shown in Figure 4. An Occam3 input file starts with the definition of the sub-component variables, which are defined as `name, cardinality, input/output, short_name`. For example, for variable `b1`, representing the first bird model component, there are four possible values, hence a cardinality of 4, and the variable is an input value, hence value 1. Observations about the values of the variables follow after the `data:` token.

```
:nominal              :data
b1,4,1,a            1 3 2 1 1 2 2 2 2 1 1
b2,4,1,b            1 2 2 1 1 1 2 2 2 1 1
b3,4,1,c            1 2 1 1 1 1 1 1 1 1 1
b4,4,1,d            1 1 1 1 1 1 1 1 1 1 1
b5,4,1,e            1 1 1 1 1 1 1 1 1 1 1
b6,4,1,f            1 1 1 1 1 1 1 1 1 1 1
b7,4,1,g            1 1 1 1 1 1 1 2 1 1 2
b8,4,1,h            1 2 1 1 1 1 2 2 2 2 2
b9,4,1,i            1 2 2 1 2 1 2 3 2 2 2
b10,5,1,j            1 2 2 2 2 2 3 3 3 3 3
cm,4,2,k             ...
```

Figure 4: Occam3 Input File.

The input file is then uploaded to Occam3, which searches for the model, in terms of variables and their values, that best defines the variables that describe the system, e.g., the centre of mass of the flock (CM). As it can be seen in Figure 5, the best results identified by Occam3 are `ABCDEGHK`, `ABCDFGHK`, and `ABCDGHIK`, representing the interaction between components `A, B, C, D, E, G, H`; `A, B, C, D, F, G, H`; and `A, B, C, D, G, I, H` respectively. It is important to highlight that these Occam3 results have been shown to reflect interaction (Willet and Zwick 2004). In other words, it is the interaction of seven out of the ten birds that leads to the observed flocking behavior represented by the values of CM.



ID	MODEL	Level	H	dDF	dLR	Alpha	Inf	%dH(DV)	dAIC	dBIC
22	IV:ABCDEGHK	7	4.0842	49149	48.9627	1.0000	1.00000000	100.0000	-98249.0373	-147188.2828
21	IV:ABCDFGHK	7	4.0842	49149	48.9627	1.0000	1.00000000	100.0000	-98249.0373	-147188.2828
20	IV:ABCDGHIK	7	4.0842	49149	48.9627	1.0000	1.00000000	100.0000	-98249.0373	-147188.2828
19	IV:ABCDGHK	6	4.0842	12285	48.9627	1.0000	1.00000000	100.0000	-24521.0373	-36753.6082
18	IV:ABCEGHK	6	4.0842	12285	48.9627	1.0000	1.00000000	100.0000	-24521.0373	-36753.6082
17	IV:ABCFGHK	6	4.0842	12285	48.9627	1.0000	1.00000000	100.0000	-24521.0373	-36753.6082
16	IV:ABCGHK	5	4.0842	3069	48.9627	1.0000	1.00000000	100.0000	-6089.0373	-9144.9396
15	IV:ACDGHK	5	4.0842	3069	48.9627	1.0000	1.00000000	100.0000	-6089.0373	-9144.9396
14	IV:ACEGHK	5	4.0842	3069	48.9627	1.0000	1.00000000	100.0000	-6089.0373	-9144.9396
13	IV:ACGHK	4	4.0842	765	48.9627	1.0000	1.00000000	100.0000	-1481.0373	-2242.7725
12	IV:BCGHK	4	4.0842	765	48.9627	1.0000	1.00000000	100.0000	-1481.0373	-2242.7725

Figure 5: Micro-Macro Relationships Using Reconstructability Analysis.

To further validate this, we removed the second rule, i.e., alignment, from the definition of the individual birds. Our results suggest that in this case there is no relationship between the micro level and macro level. This is also confirmed visually, in that no flocking behavior occurs. Space constraints prevent us from showing this visualization here.

4.2 Live Emergence Analysis

We define a bird model objective as: $o(b_i) = (description, var(o(b_i)))$ where

$$description = \textit{Fly northbound with an average speed of 20km/hour}; var(o(b_i)) = \{direction, speed\}$$

For simplicity and for alignment with the previous example, we discuss a simplified case where a) the speed of each bird is calculated based on the distance from the center of the drawing panel; and b) the calculated state S_{rk} is calculated at each simulation step by considering the reunion of bird states when each bird in is executed in *isolation*, i.e., without interacting with other birds in the flock. This follows the grammar-based approach as proposed in Kubik (2003) but limits the study of systems where individuals can also function independently, as is the case for well-known examples of emergence such as flocks of birds and the game of life. The study of calculating S_{rk} without this assumption is part of our future work.

If each individual is executed independently, the direction of flight does not change and remains northbound (e.g., "N" in Table 2(a)). However, the value of the distance increases rapidly and each individual bird leaves the panel within four simulation steps (e.g., "-" in Table 2(a)). Table 2(a) and 2(b) present the collated results for the first five birds.

Table 2: Observation of the Flock of Birds.

(a) Independent Individuals

Bird 1	Bird 2	Bird 3	Bird 4	Bird 5
120, N	114, N	135,N	111, N	98, N
155, N	193, N	146, N	180, N	150, N
360, N	394, N	387, N	380, N	350, N
539, N	571, N	560, N	558, N	538, N
-, N	-, N	-, N	-, N	-, N
-, N	-, N	-, N	-, N	-, N
-, N	-, N	-, N	-, N	-, N
-, N	-, N	-, N	-, N	-, N
-, N	-, N	-, N	-, N	-, N
-, N	-, N	-, N	-, N	-, N

(b) Entire Group

Bird 1	Bird 2	Bird 3	Bird 4	Bird 5
53, S	209, N	128,N	67, S	43, N
45, S	174, NW	112, N	65, S	44, N
40, S	129, NW	88, N	60, SW	43, N
39, S	76, NW	59, N	51, SW	38, N
38, N	27, NW	42, N	34, NW	37, N
54, N	32, NW	48, N	11, NW	57, N
59, N	71, NW	69, NW	13, NW	75, N
56, NW	113, NW	100, NW	42, NW	94, N
58, NW	154, NW	136, NW	79, NW	118, NW
82, NW	195, NW	175, NW	125, NW	151, NW

Step 1. Calculation of S_{rk}
In the case where individual birds are considered in isolation, and the measure of statistical difference looks at distances greater than $\varepsilon = 150$, $S_{rk} = \{S_{r1}, S_{r3}, S_{r4}\}$, where $S_{r1} = \{(120,N)(111,N),(114,N),(111,N)(135,N),(111,N),(98,N)\}$, $S_{r3} = \{(360,N)(111,N),(394,N),(111,N)(387,N),(111,N)(380,N),(111,N)(350,N)\}$, $S_{r4} = \{(539,N),(571,N),(111,N)(560,N),(111,N)(558,N),(111,N)(538,N)\}$, i.e., the first, third, and fourth line in Table 2(b).

Step 2. Observation of S
S represents the reunion of all the system states, as $S = \{S_i | i = 1 \dots 10\}$, representing each line in Table 2(a) respectively.

Step 3. Live Emergence Identification
It is evident that the states representing the data in Table 2(a) are different from the states in Table 2(b), with the most significant differences appearing when flocking occurs. This is because without flocking, there are still individuals that follow their own individual flight path, similar to the case when they are considered independently. In particular, for $\varepsilon = 150$ for the distance attribute, and a similar threshold to distinguish directions with respect to their cardinal point, e.g., N and NW are closer than N and S, the most significant difference occurs between S_{r4} and $S4$.

5 CONCLUSION

This paper proposes an approach for the identification of emergent behavior in component-based simulation models with two orthogonal perspectives. In *post-mortem emergence* analysis, the emergent behavior has been observed, and the purpose is to identify the causes, in terms of model components states and interactions, that led to the emergent behavior. In *live emergence* analysis, the emergent behavior is identified as the simulation is executed, as any simulation state that deviates from a calculated composed model state. We exploit reconstructability analysis in both post-mortem and live analysis, to identify significant interactions between components that leads to emergence and to obtain a calculated system state that is then compared to the simulated state respectively.

Our approach is a first step towards the automated identification and validation of emergent behavior in complex systems. Our initial experiments on using a flock of birds model highlight the feasibility of our proposed approach and its applicability to more complex models, but also important points for future work. First, reconstructability analysis requires several observations of the composed model in order to determine the relevant variables for reconstruction. Towards this, future work includes applying our proposed conceptual method to various real-life examples such as road traffic and social networks models. Another important issue remains how to leverage on the existing body of work in the classification of various emergent properties, by properly annotating and classifying properties in our proposed emergent property repository. Lastly, while we propose the visualization of emergent behaviors to increase understanding, the question of whether a particular emergent behavior is harmless or detrimental remains an open challenge.

Acknowledgments

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

REFERENCES

Bedau, M. 1997. "Weak Emergence". *Philosophical Perspectives* 11:375399.

Cavallo, R., and G. Klir. 1979. "Reconstructability Analysis of Multi-dimensional Relations: A Theoretical Basis for Computer-aided Determination of Acceptable System Models". *Int. Journal of General Systems* 5:143171.

Chan, W. K. V. 2011, December. "Interaction Metric of Emergent Behaviors in Agent-based Simulation". In *Proceedings of the 2011 Winter Simulation Conference*, edited by S. Jain, R. R. Creasey, J. Himmelspach, K. P. White, and M. Fu, 357–368. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Chen, C., S. B. Nagl, and C. D. Clack. 2007. "Specifying, Detecting and Analysing Emergent Behaviours in Multi-Level Agent-Based Simulations". In *Proceedings of the Summer Computer Simulation Conference*.

Chen, C. C., S. B. Nagl, and C. D. Clack. 2009. "A Formalism for Multi-level Emergent Behaviours in Designed Component-based Systems and Agent-based Simulations". *Understanding Complex Systems*.

Chi, L., W. K. Chan, G. Seow, and K. Tam. 2009. "Transplanting Social Capital to the Online World: Insights from Two Experimental Studies". *Journal of Organizational Computing and Electronic Commerce* 19:214–236.

Fayyad, U., and R. Uthurusamy. 2002. "Evolving Data Into Mining Solutions for Insights". *Communications of the ACM* 45.

Floyd, S., and V. Jacobson. 1993. "The Synchronization of Periodic Routing Messages". In *Proceedings of SIGCOMM*, 33–44.

US-Canada Power System Outage Task Force 2004. "Final Report on the August 14, 2003 Blackout in the United States and Canada".

Gardner, M. 1970. *The Fantastic Combinations of John Conways New Solitaire Games*. Mathematical Games.

Gore, R., and P. Reynolds. 2007, December. "An Exploration-based Taxonomy for Emergent Behavior Analysis in Simulation". In *Proceedings of the 2007 Winter Simulation Conference*, edited by S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 1232–1240. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Gore, R., and P. Reynolds. 2008, December. "Applying Causal Inference to Understand Emergent Behavior". In *Proceedings of the 2008 Winter Simulation Conference*, edited by S. J. Mason, R. R. Hill, L. Moench, O. Rose, T. Jefferson, and J. W. Fowler, 712–721. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Holland, J. 1999. *Emergence, From Chaos to Order*. Basic Books.

Johnson, C. W. 2006. “What are Emergent Properties and How Do They Affect the Engineering of Complex Systems?”. *Reliability Engineering and System Safety* 12:1475–1481.

Jones, B. 1985. “A Greedy Algorithm for a Generalization of the Reconstruction Problem”. *Int. Journal of General Systems* 11:63–68.

Kubik, A. 2003. “Towards a Formalization of Emergence”. *Journal of Artificial Life* 9:41–65.

M. Zwick Last retrieved Apr. 2012. “OCCAM”. http://dmm.sysc.pdx.edu/weboccam.cgi?action=search.

Mogul, J. C. 2006. “Emergent (mis)behavior vs. Complex Software Systems”. In *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems*, edited by W. Zwaenepoel, 293–304. New York, USA.

Page, E., and J. Opper. 1999, December. “Observations on the Complexity of Composable Simulations”. In *Proceedings of the 1999 Winter Simulation Conference*, edited by P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. Evans, Volume 1, 553–560. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Ramakrishnan, K. K., and H. Yang. 1994. “The Ethernet Capture Effect: Analysis and Solution”. In *Proceedings of the IEEE Local Computer Networks Conference*, edited by H. Yang, 228–240. Minneapolis, USA.

Reynolds, C. 1987. “Flocks, Herds, and Schools: A Distributed Behavioral Model”. In *Proceedings of ACM SIGGRAPH*, edited by M. C. Stone, 25–34.

Seth, A. K. 2008. “Measuring Emergence via Nonlinear Granger Causality”. In *Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, edited by T. J. Hutton, 545–553.

Srba, J. 2001. “On the Power of Labels in Transition Systems”. In *Proceedings of the 12th International Conference on Concurrency Theory*, edited by K. G. Larsen and M. Nielsen, 277–291. Aalborg, Denmark.

Szabo, C., and Y. Teo. 2012a. “An Objective-based Approach for Semantic Validation of Emergence in Component-based Simulation Models”. In *Proceedings of the 26th ACM/IEEE/SCS Workshop on Principles of Advanced and Distributed Simulation*, (to appear). Zhangjiajie, China.

Szabo, C., and Y. Teo. 2012b. “Semantic Validation of Emergent Properties in Component-basedSimulation Models”. *Ontology, Epistemology, and Teleology of Modeling and Simulation Philosophical Foundations for Intelligent M&S Applications*:319–333.

Szabo, C., Y. Teo, and S. See. 2009, December. “A Time-based Formalism for the Validation of Semantic Composability”. In *Proceedings of the 2009 Winter Simulation Conference*, edited by M. D. Rossetti, R. R. Hill, B. Johansson, A. Dunkin, and R. G. Ingalls, 1411–1422. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Teo, Y., and C. Szabo. 2008. “CODES: An Integrated Approach to Composable Modeling and Simulation”. In *Proceedings of the 41st Annual Simulation Symposium*, edited by H. Karatza, 103–110. Ottawa, Canada.

Willet, K., and M. Zwick. 2004. “A Software Architecture for Reconstructability Analysis”. *Kybernetes* 33:97–1008.

AUTHOR BIOGRAPHIES

CLAUDIA SZABO is an Associate Lecturer at the School of Computer Science, The University of Adelaide in Australia. Her email address is claudia.szabo@adelaide.edu.au.

YONG MENG TEO is an Associate Professor with the Department of Computer Science at the National University of Singapore, and a Visiting Professor at the Shanghai Advanced Research Institute, Chinese Academy of Sciences in China. He heads the Computer Systems Research Group and the Information Technology Unit. His email address is teoym@comp.nus.edu.sg.