Specifying, Detecting and Analysing Emergent Behaviours in Multi-Level Agent-Based Simulations

Chih-Chun Chen, Sylvia B. Nagl and Christopher D. Clack
University College London, Gower Street, London, WC1E 6BT, UK
c.chen@cs.ucl.ac.uk, s.nagl@medsch.ucl.ac.uk, clack@cs.ucl.ac.uk

Abstract

We introduce a method for analysing emergent behaviours in multi-agent simulations using *complex events*. Complex events are composed of interrelated events, and they can be defined at any level of spatio-temporal abstraction (equal to or above the lowest level of abstraction given by the model). Minimal *types* of complex events define sets, which are equated with particular emergent behaviours and can be detected in simulation.

Since complex events are derived from the agent-based model itself, they provide significant benefits when compared with traditional state-aggregation methods. First, they provide a method of *specifying* emergent behaviour, so that such behaviour can be monitored. Second, they provide a mechanism that *retains the underlying structure* of that behaviour. This latter property supports analysis of the mechanisms at lower levels that give rise to emergent behaviours, and identification of patterns between levels. In other words, multi-agent simulations become less 'opaque'[1].

1. INTRODUCTION

Emergent behaviour in real-world systems is notoriously unpredictable. When the UK government ordered the National Health Service (NHS) to reduce the average time that patients spent waiting for treatment, the emergent behaviour at some hospitals was unexpected: quick operations were prioritised over longer operations; patients were given unnecessary treatment to move them down the waiting list; a new role of "hello nurse" was invented to greet patients (so they were no longer "waiting").[1] It would surely be helpful if we could model and simulate a complex system such as the NHS, describe and observe emergent behaviours, and investigate those behaviours to understand *how* they were created (i.e. what were the constituent components or behaviours at a lower level of abstraction that caused the behaviour to emerge at the higher level?). However, it is difficult to model the behaviour "prioritising a quick operation over a longer operation" since the more interesting instances have many sub-

---

[1] http://www.blairwatch.co.uk/node/1692
http://www.24dash.com/health/19024.htm

components distributed across a significant timescale.

In this paper we introduce a new method for specifying, detecting and analysing emergent behaviours that preserves their underlying and contributory structure.

Agent-based simulations have been widely used in many domains to study dynamic systems which also have dynamic structures. Rules are defined at the level of agents, with the behaviour of each type of agent being governed by its own set of rules. From these agent-level behaviours, certain properties can emerge at the 'macro-level'; these can be atemporal patterns/structures or temporally extended properties, i.e. behaviours. While atemporal emergent patterns can be described and analysed in terms of configurations of agent states, there seems to be no corresponding structure-preserving method for doing this for temporally extended emergent properties. Instead, two types of approaches are typically used to measure 'macro-level' behaviour:
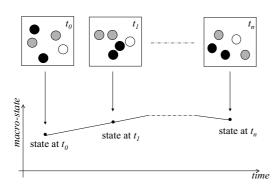
1. The aggregation of agent states ("State Aggregation") into a macro-variable or set of macro-variables (see [2, 3] for examples). Changes in the macro-variable(s) over the course of the simulation represent changes at the macro-level (see Figure 1).

2. Human observation of the simulation, and of qualitative changes in the dynamics e.g. flocking behaviour from non-flocking behaviour [4] (a method for quantifying this is given in [5]).

The use of State Aggregation results in loss of information about the *structure* of behaviour e.g. which agent interactions have given rise to the behaviour, and how these interactions are related in time and space. Since a set of states is aggregated into a single measure, the relationship between the high-level behaviour and the underlying agent behaviours (generated from the agent-based model (ABM)) is lost. This lack of understanding as to how the agent level relates to higher levels of observation is what makes simulations 'opaque' [1]. On the other hand, human observation of the simulation does preserve structure (which structures are identified depends very much on the expertise and objectives of the observer), but as a scientific method it is insufficiently precise or methodical.

What we lack is a systematic method for describing emergent behaviours in terms of the agent-based model. In some

Figure 1. Caricature of the traditional view of multi-agent simulation as a series of still frames. Each frame is a snapshot of the system at a given time.

cases, this greatly hampers the contribution that ABM can make to our understanding of a complex system. For example, one might view the higher-level emergent behaviours of the NHS as a consequence of its members' behaviours (doctors, managers, consultants, staff, patients, etc.). We might design and run a multi-agent simulation of this successfully which produces the expected changes in the system's overall performance. However, if this change in overall performance is represented by a single variable, we lose a substantial amount of useful information even though it is in our simulation. We fail to identify the underlying agent interactions and organisational behaviours that give rise to the higher-level aberrant behaviour. How does bad management contribute to underperformance? Are operational inefficiencies important? We fail to identify these emergent behaviours in a simulation because they not been described in terms of what is going on at the agent level. Yet answers to these types of questions give us important information about the workings of the system and which events are most significant in causing failure.

We propose a method for addressing this issue of how to specify, detect and analyse emergent behaviours in multi-agent simulations *in terms of the agent-based model*. The method applies to any level of spatio-temporal abstraction that can be composed from the agent-based model and hence allows us to relate different levels to one another using the underlying agent model as a common denominator. Using a simple example, we will demonstrate that our new method supports an improved understanding of emergent behaviour that cannot be provided by State Aggregation.

1.1. Structure of the Paper

Since our work is based on certain assumptions about complex systems and emergence, those assumptions are made explicit in Section 2. We also give a brief generalised description of agent-based models.

Section 3. explains how events can be composed to give complex events by specifying temporal, spatial and component-based relationships between constituent events, and Section 3.2. shows how emergent behaviours can be identified with *sets* of complex events. This provides a systematic method for specifying temporally extended emergent properties (behaviours and entities) in terms of the underlying agent-based model.

To illustrate our method, we give a simple worked example and then run simulations to demonstrate the detection and decomposition of of emergent behaviours. The final section briefly summarises and concludes.

2. DEFINITIONS AND ASSUMPTIONS

The definition of terms such as 'emergence' and 'levels' is heavily contested in the literature [6, 7]. In order to make explicit our assumptions therefore, we first give our definitions of the emergence-related terms used throughout the paper. We then give a generalised description of agent-based models.

2.1. Emergence and Emergent Properties

In this paper, we assume a particular definition of emergence, based largely on the recent contributions from statistical mechanics that have sought to formalise emergence (see [8] for a review); in particular, we use many of the definitions given in [9]. While we acknowledge that these definitions will not be accepted by all who work in the field, we argue that for our current purposes they are useful definitions and provide a sound framework for defining emergence in multi-agent simulations. For our assumed definitions, we must first introduce the concepts of scope and resolution.

The **scope** of a representation of the system is the 'set of components within the boundary between the associated system and its environment' [9] [2] (at a given resolution). The scope S of a temporally extended system can be considered to be made up of its temporal scope S_τ, which defines the set of moments of time over which the system is represented and a spatial scope S_x.

Resolution is the number of states that can be distinguished i.e. given the same scope, a higher-resolution (finer) representation will be able to distinguish a greater number of possibilities. Again, there is both a spatial aspect R_x and a temporal aspect R_τ which together define the overall resolution R.

A level of abstraction is then a function of the scope and resolution, where a higher level of abstraction M has a greater scope and/or a lower resolution than a lower level of abstraction μ (see equations (1), (2) and (3)):

$$R_M \leq R_\mu \tag{1}$$

---

[2]The system's environment is considered to be outside the scope of representation.

$$S_M \geq S_\mu \qquad (2)$$

$$(R_M, S_M) \neq (R_\mu, S_\mu) \qquad (3)$$

Emergence is closely related to level of abstraction. Firstly, we stipulate that an **emergent property** must be at a higher level of abstraction than its constituent properties. Secondly, an emergent property must not be detectable at a lower level of abstraction (note that this is different from saying that it can not be detected *by* lower level components). Finally, the reason why an emergent property is not detectable at lower levels is because it consists of a particular set of relationships between its components as well as the components itself.

We further distinguish three types of emergent property:

1. emergent **state**: an identifiable state at a particular level of abstraction that results from a configuration of states at lower levels of abstraction, defined atemporally[3].

2. emergent **entity**: an identifiable entity at a particular level of abstraction that is able to persist through time (has temporal extension) and which is subject to the rules operating at that level of abstraction, but whose existence is dependent on entities and/or processes at lower levels of abstraction.[4]

3. emergent **behaviour**: an identifiable temporally extended process at a particular level of abstraction that results from a set of processes operating at a lower level of abstraction that are *related to one another* temporally and/or spatially.

2.2.   Generalised Description of ABMs

An agent-based model (ABM) consists of a set of specifications for different component types, a **component** being any element in the system that can be uniquely identified and that can persist through time i.e. has an identity and is able to have a history. Various classes of components (e.g. agents, objects, unencapsulated state variables) might exist in the model; although by definition agents should always be present, non-agent component classes might also feature. Specifications for component types define the set of possible states that instances of the type can take, and are usually given in the form of rules governing state transitions. A **state transition rule** is a function that changes the values of a set of variables when a particular condition is satisfied. Agents and objects are said to *encapsulate* a set of variables, which together represent the state of that agent or object. Most of the state transition rules

in an agent-based model are specified from the perspective of a given component type.[5]

3.   COMPLEX EVENTS

In this section we introduce our new concept of *complex events*, which are events that comprise one or more related constituent events (the constituent events can also be complex events). Thus, we define a complex event *CE* to be either a simple event *SE* (with no further constituents) or two complex events linked by a specified relationship $\bowtie$:

$$CE \;::\; SE \mid CE_1 \bowtie CE_2 \qquad (4)$$

Relating Two Complex Events

We define the relationship $\bowtie$ to be a temporal operator $\otimes$ optionally followed by descriptions of (i) space constraints and (ii) constraints pertaining to the variables or components of the two related complex events. A detailed explanation of $\otimes$ and the two types of constraint is provided in Section 3.1.

Simple Events

We define a simple event *SE* as a change in state (given by a state transition function *trans*()) that occurs in time with a non-negative duration (duration ≥ 0) — see Equation 5, where $x_{id_1}, \ldots x_{id_i}$ are specific variables in simulation, and t_{start} and t_{end} are respectively the start and end times of the simple event. The variables may belong to different components (e.g. agents), and the source component identity for each variable may be specified as part of the subscripts id_1, id_2 etc.

Each simple event is an instance of a *simple event type T_{SE}*, which specifies (i) a transition function to be applied to specified variable types, and (ii) a duration — see Equation 6, where t_i are the types of the variables and d denotes the duration (a range, or defined value). An event type specifies a set while an actual event is a member of such a set. Two simple events are therefore of the same type (members of the same set) if they apply the same function to the same variable types for the same duration.

$$SE \;::\; (trans(x_{id_1}, \ldots x_{id_i}), \; [t_{start}, t_{end}]) \qquad (5)$$

$$T_{SE} \;::\; (trans(t_1, \ldots t_i), \; d) \qquad (6)$$

State Transition Rules

In Section 2.2. we defined a state transition rule as a function that changes the values of a set of variables when a particular condition is satisfied. Thus Equation 7 gives the syntax

---

[3] An atemporal definition means one that does not *internally* refer to time e.g. being blue. So while we can talk about states with reference to time e.g. say that a state persists in time, the description of the state *itself* does not include a reference to time.

[4] Note that the *state of an emergent entity* is not the same as an *emergent state*.

[5] E.g. a typical rule governing agent 'actions' has the format: if an agent instance of type *A* 'perceives' that condition *x* is satisfied in its 'neighbourhood', transition *f* occurs, with *x* and *f* being defined relative to the *A* instance rather than to some global frame of reference (an analogy can be drawn with local and global coordinates in computer graphics). The modified variables may be encapsulated by zero or more components.

for a state transition rule STR, where C is the condition that needs to be satisfied for an event of type T_{SE} to occur:

$$STR :: (T_{SE}, C) \qquad (7)$$

An event e of type E can be the condition for another event f of type F if the set of variable values that results from an event of type E is always equal to the condition for an event of type F, as defined by some state transition rule i.e. if

$$(C_f == trans_e), \forall e \in E, \forall f \in F$$

On the other hand, an E-type event e_i can be a condition for an F-type f_j even when this relationship does not hold; i.e. when the result of applying $trans_{e_i}$ contingently results in variable values that form the condition for another event. In a dynamic simulation, the effect of a given event has different consequences depending on previously and concurrently occurring events; i.e. depending on context (both spatial and temporal), the effect of an event can lead to different subsequent events ($trans_e == C_f$ in some contexts and $trans_e == C_g$ in others).

3.1. Specifying Complex Events

The concept of event composition has previously been introduced in related work in the context of ABM e.g. [10, 11] but such compositions mainly relate to individual *agent* behaviours; the approach presented here builds on this work but is more general since it addresses other relationships between events besides temporal ones.

In Section 3. we defined a complex event as being either a simple event or a relationship between two complex events, i.e. $C_1 \bowtie C_2$. As stated previously, we define the relationship $\bowtie$ to be a temporal operator $\otimes$ optionally followed by descriptions of (i) space constraints and (ii) constraints pertaining to the variables or components of the two related complex events. Thus, the syntactic pattern for a complex event relationship $\bowtie$ is given in Equation 8.

$$e_1 \bowtie e_2 :: e_1 \otimes [space] [var] e_2 \qquad (8)$$

where

- **The temporal operator '$\otimes$':** may for example specify that the second event e_2 is initiated at the same time '$\|$', before '$\prec$', after '$\succ$', or immediately after ';' the first event e_1;

- **The spatial constraint '$space$':** defines the space within which e_2 should occur relative to e_1; and

- **The component or variable constraint 'var':** defines the relationships between variables or components of the two events e_1 and e_2.

$\otimes$, $[space]$ and $[var]$ can be specified in any number of ways using expressions derived from different systems of logic and representation, depending on the expressivity requirements of the specifications. To give an example, we introduce the token identity operator '/', which is a $[var]$ operator. To specify which variable or component instances are shared between two events, it is necessary to 'get inside' the state transition functions[6]. The expression follows one of two patterns:

$$(trans_{e_1}(vars), trans_{e_2}(vars), [e_1(var)/e_2(var)])$$

or

$$(src.trans_{e_1}, src.trans_{e_2}, [(src)e_1/(src)e_2)])$$

The first pattern constrains the variable *var* in e_1 to be *the same* as the variable *var* in e_2 — e.g. their subscripts must show that they are encapsulated by the same component (agent) and they must have the same name. *vars* stands for the variables involved in each state transition function.

The second pattern uses the syntax $src.trans_{e_1}$ to denote the source component (agent) that gave rise to the transition $trans_{e_1}$ (via event e_1, as the result of some state transition rule). The token identity operator expression $(src)e_1/(src)e_2$ then constrains the source component (agent) for event e_1 to be *the same* as the source component (agent) for event e_2.
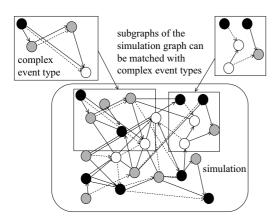
Complex Event Types

A complex event type T_{CE} is specified by specifying the constituent event types and the relationships that must be satisfied between instances of these types. More than one instance of an event type can be required in the complex event so that when a complex event type is instantiated, each event instance plays a particular role in the complex event.

We can represent a complex event type specification as a directed multi-graph (see Figure 2). A multi-graph is a set of nodes N, indexed by integers and a directed adjacency relationship $arc(n1, n2)$ defined for pairs of distinct nodes and returning a non-negative integer value. The arcs are categorised by colour (the colours are greyscales in the figure) to support more than one type of adjacency relationship i.e. $\{arcA(n1,n2), arcB(n1,n2), arcC(n1,n2)...\}$, each of which may or may not be satisfied between two nodes i.e. returns $true$ or $false$. This means that no more than one instance of an adjacency relationship type (arc colour) (arc shade) can exist between two nodes. A colour function is a function $colour(n)$ associated with a set of nodes; for each node n, $colour(n)$ is the colour of that node.

The nodes represent the event instances in the complex event type, with the colours standing for the event types. The

---

[6]Note that if distinct instances of a type of variable are stipulated inside a transition function e.g. x_0 and x_1, this must be preserved. The operator '/' only equates variable or component instances within *different* events.

arcs represent directed relationships between two events with colours standing for particular types of relationship e.g. $\prec$, $distance = 3$. A simulation can also be represented as a directed multi-graph with coloured nodes and coloured arcs (many of whose colours we have no knowledge of). We can therefore identify instances of complex events by identifying subgraphs in the simulation graph which are isomorphic with the complex event type graphs.[7] This provides us with a means of detecting the complex event types we have specified when they are instantiated in simulation.

Figure 2. A simulation can be represented by a coloured multi-graph with coloured arcs. Different node colours (represented here by different shades of grey) stand for the different event types. Arcs represent relationships (some of which are of no interest to the modeller) with the colours (represented here by different line types — solid and dashed) standing for different relation types. We say that a complex event type is instantiated in simulation when we can identify a subgraph in the simulation graph that is isomorphic with the complex event type graph.

3.2. Specifying Emergent Behaviour with Sets of Complex Event Types

Obviously, it is not feasible to specify all the possible complex event types that model an emergent behaviour or emergent entity, since these can be realised in many different ways (many of which we have no explicit knowledge of). Rather, we should be able to define classes or sets of complex event types that model an emergent behaviour. Each member of a set then represents an exemplar of a particular emergent behaviour. Each Complex Event must contain at least one complex event type T_{CE} that is *minimal* — i.e. they capture the minimal behaviour that can be categorised as an example of

---

[7]Note, also that a given component can participate in many graphs at the same time. This equates to its participation in different events at various levels of abstraction.

the emergent behaviour described by the set.[8]

For example, the set of minimal complex event types for flocking behaviour are all those where two boids within a defined spatial distance d from one another move through space in the same direction over some minimum length of time t. So we can say that an emergent behaviour is modelled by a set of minimal complex event types and instantiated in simulation when one of these minimal complex events is instantiated (see Figure 3). Any further complexity that may be required for specifying or detecting emergent behaviour may be incorporated via the specification of more elaborate complex events.

Although we use crisp sets in the current discussion, our scheme can be extended to incorporate fuzzy set concepts [12, 13] by allowing different degrees of membership for both complex event type sets and emergent behaviour sets.
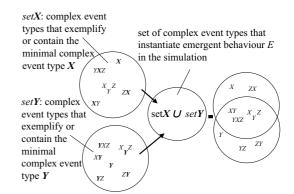

Figure 3. The emergent behaviour E is said to be instantiated in a simulation when an event of one of the event types in the set *setX* or *setY* is instantiated. The set of complex event types that represent E is the union of the sets *setX* and *setY*.

4. EXAMPLE: PREDATOR-PREY MODEL

To demonstrate our method for understanding simulations in terms of complex events and emergent behaviours, we use a simple predator-prey model. However, instead of only considering overall population changes, we also try to understand the way these come about through emergent group behaviours by detecting the the complex events associated with them.

4.1. Model: Agent Rules and Validation

In our model, there are two species: lions and antelopes. Lions are the predator species and antelopes the prey.

The simulation rules for a lion, and associated state transition rules, are:

---

[8]The minimal complex event type specifications themselves define a set of event types.

(*l=lion*; *a=antelope*; *x* and *y* are coordinates)

1. If an antelope is detected within distance d, kill antelope with probability $p(kill)$. The killing of an antelope takes place instantly.

   $(KillA, C_{killA})$
   where
   $KillA = (trans_{KillA}(a, (x, y)), 0)$
   $trans_{KillA}(a, (x, y)) = dead(a)at(x, y)$
   $C_{KillA} = exists(a)within(d) \wedge (random \leq p(kill))$

2. If the antelope is killed, a new lion is born at the location of the dead antelope (which is then removed from the system). The birth of a new lion takes place after a delay of one time step.

   $(LBirth, C_{LBirth})$
   where
   $LBirth = (trans_{LBirth}((x, y)), 1)$
   $trans_{LBirth}((x, y)) = replace(l, a)at((x, y))$
   $C_{LBirth} = deadA((x, y))$

3. Die with probability $p(lDeath)$ (instantly) if there is no antelope within distance d.

   $(LDeath, C_{LDeath})$
   where
   $LDeath = (trans_{LDeath}(l, (x, y)), 0)$
   $trans_{LDeath}(l, (x, y)) = empty(x, y)$
   $C_{LDeath} = (\neg C_{KillA}) \wedge (random \leq p(LDeath))$

4. Move one step in a random direction at each time step if no killing of antelopes or death has taken place.

   $(MoveL, C_{MoveL})$
   where
   $MoveL = (trans_{MoveL}(l, (x, y))), 0)$
   $trans_{MoveL}(l, (x, y)) = Move(l)to(x_1, y_1); new(x_1, y_1)$
   $C_{MoveL} = (\neg C_{KillA}) \wedge (\neg C_{LDeath})$

   The simulation rules and state transition rules for an antelope are:

1. Move one step in a random direction at each time step (if not dead).

   $(MoveA, C_{MoveA})$
   where
   $MoveA = (trans_{MoveA}(a, (x, y))), 0)$
   $trans_{MoveA}(a, (x, y)) = Move(a)to(x_1, y_1); new(x_1, y_1)$
   $C_{MoveA} = any$

2. A new antelope is born at random location with probability $p(aBirth)$

   $(ABirth, C_{ABirth})$
   where
   $ABirth = (trans_{ABirth}((x, y)), 0)$
   $trans_{ABirth}((x, y)) = new(a)at(x_1, y_1); random(x_1, y_1)$
   $C_{ABirth} = random \leq p(aBirth)$

4.2. Experiment: What are the Mechanisms Underlying Population Dynamics?

To illustrate the specification and detection of emergent behaviours using complex events, we introduce two different emergent behaviours, *between_lion_overhunting* and *same_lion_overhunting* to give us insight into the mechanisms underlying changes in lion population. While tracking the numbers of lions and antelopes would permit measurement of overall system behaviour, this overall system behaviour can have different underlying causes. We distinguish between *between_lion_overhunting*, where more than one lion within a particular area makes a kill resulting in *starvation* in the area or *same_lion_overhunting*, where a single lion kills twice in immediate succession and then starves.

In our experiment, we ensure the lions will become extinct by making them voracious killers — we set $p(kill) = 1$ so that a lion always kills when it has an antelope in its neighbourhood. Since a lion cannot die when killing an antelope, its death must be a consequence of not having an antelope nearby; we will then investigate the emergent behaviour to determine whether extinction is caused by the overhunting of single lions, or by the competition of two or more lions hunting in close proximity.

We define *starvation*, *between_lion_overhunting* and *same_lion_overhunting* behaviours as follows.

- *starvation*: A lion goes for three time steps without seeing an antelope and then dies.

$$MoveL_0; [same(l)]MoveL_1$$
$$; [same(l)]MoveL_2; [same(l)]LDeath_0$$

where *same(l)* is:

$$(trans_{MoveL_0}(l, (x, y)), trans_{MoveL_1}(l, (x, y)),$$
$$[MoveL_0(l)/MoveL_1(l)])$$

- *between_lion_overhunting*: Two different lions within a given distance *range*1 from one another each kill an antelope (either the same or different antelope) at the same

time ('||'). After this, there is at least one starvation event in the area inhabited by these two lions (*range*2).

$$(l.KillA_0 || [within(range1)][\neg(l)same]KillA_1$$
$$; [within(range2)]starvation)$$

- *same_lion_overhunting*: The same lion kills an antelope two time steps in succession and then dies from starvation.

$$l.KillA_0; [(l)same]l.KillA_1$$
$$; [(l)same]starvation$$

where (*l*)*same* is:

$$(trans_{l.KillA_0}(a,(x,y)), trans_{l.KillA_1}(a,(x,y)),$$
$$[(l)KillA_0/(l)(KillA_1)])$$

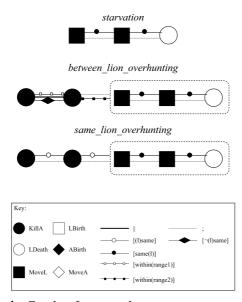Figure 4 shows how these complex event types can be represented by graphs.

Figure 4. Graphs for complex event types *starvation*, *between_lion_overhunting* and *same_lion_overhunting*. With key below.

Figure 5 shows the degree to which the above complex event types occur in simulations with different population densities before the extinction of lions. In this particular example, we use a simple additive measure that counts the number of times the above complex event types are detected in the simulation. Notice that *same_lion_overhunting* occurs very

infrequently. This is likely to be due to the low antelope numbers, which means that it is unlikely for the same individual to make a kill twice. Also, while *same_lion_overhunting* is limited by the number of individuals, *between_lion_overhunting* is not, since the same individual can participate in more than one instance of *between_lion_overhunting*.

Both *between_lion_overhunting* and *same_lion_overhunting* complex event types define a particular set of relationships between events that are temporally and spatially structured and which can be detected in simulation. By contrast, State Aggregation methods lose information about the deep structure of emergent behaviour. Also, assigning variables to particular quantities, e.g. number of agents and number of deaths, does not enable us to identify or quantify emergent *behaviours* since the variables are based on system *states* at different points in time and hence only measure the *consequences* of behaviours at particular times. On the other hand, human observation of the simulation does not relate directly to the model. Because complex events are defined ultimately in terms of the agent-based model itself (its events and rules), we are able to understand how model rules relate to multi-level behaviours and how alterations in these rules are likely to affect these behaviours.

Furthermore, we can define complex events at different degrees of generality and further investigate their more detailed structures. For example, *between_lion_overhunting* can be subclassed into *same_antelope* and *different_antelope*. Once we have identified a set of complex events belonging to the same type, we can specify further constraints to distinguish between these events.
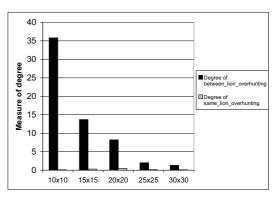

Figure 5. Graph showing the degree of *same_individual_overhunting* and *between_individual_overhunting* for different population densities (controlled by grid area). $p(kill) = 1$, $numLions = 50$, $numAntelopes = 10$, $p(aBirth) = 0$, $p(lDeath) = 0.1$. The smallest grid size (highest density) gives the greatest number of instances of *between_individual_overhunting*.

5. SUMMARY AND CONCLUSIONS

In this paper we have introduced and demonstrated a novel method using Complex Events for identifying specific emergent behaviours in an agent-based simulation. This can be applied to behaviour at any level of abstraction above the agent-based model level.

Since emergent behaviours are composed of events at the model level, we do not have to lose information about the structure of behaviours at different levels (unlike previous State Aggregation methods). The decomposition of higher level behaviours into lower level events allows us to predict more reliably how changes at the model level (e.g. changes in agent rules) affect behaviours at multiple levels.

We have demonstrated how the use of Complex Events provides a twofold benefit: it provides a method of specifying emergent behaviour, so that such behaviour can be monitored; and it provides a mechanism that retains the underlying, contributory structure of that behaviour. This therefore provides a rich mechanism for specifying, detecting and analysing emergent behaviour.

REFERENCES

[1] E. A. D. Paolo, J. Noble, and S. Bullock, "Simulation Models As Opaque Thought Experiments," in *Artificial Life VII: The Seventh International Conference on the Simulation and Synthesis of Living Systems*, (Reed College, Portland, Oregon, USA), August 2000.

[2] L. Tesfatsion and K. Judd, eds., *Handbook Of Computational Economics 2: Agent-Based Computational Economics*. Elsevier, 2007.

[3] N. Moreira, "In Pixels And In Health: Computer Modeling Pushes The Threshold Of Medical Research," *Science News*, vol. 169, pp. 40–41, January 2006.

[4] C. W. Reynolds, "Flocks, Herds And Schools: A Distributed Behavioural Model," *Comp. Graph.*, vol. 21, no. 4, pp. 25–34, 1987.

[5] W. A. Wright, R. E. Smith, M. Danek, and P. Greenway, "A Measure Of Emergence In An Adapting, Multi-Agent Context," in *Proceedings of the Sixth International Conference on the Simulation of Adaptive Behaviour, SAB-2000*, pp. 20–27, 2000.

[6] M. Christen and L. R. Franklin, "The Concept Of Emergence In Complexity Science: Finding Coherence Between Theory And Practice," in *Proceedings of the Complex Systems Summer School*, 2002.

[7] J. Deguet, Y. Demazeau, and L. Magnin, "Elements About The Emergence Issue: A Survey Of Emergence Definitions," *ComPlexUs*, vol. 3, pp. 24–31, August 2006.

[8] C. R. Shalizi, *Methods And Techniques Of Complex Systems Science: An Overview*, ch. Methods and Techniques of Complex Systems Science: An Overview, pp. 33–114. New York: Springer, 2006.

[9] A. Ryan, "Emergence Is Coupled To Scope, Not Level," *Nonlinear Sciences*, 2007.

[10] T. Stratulat, F. Clerin-Debart, and P. Enjalbert, "Norms And Time In Agent-Based Systems," in *ICAIL*, 2001.

[11] A. Burns, I. J. Hayes, G. Baxter, and C. J. Fidge, "Modelling Temporal Behaviour In Complex Socio-Technical Systems," tech. rep., University of York, 2005.

[12] L. A. Zadeh, "Fuzzy sets," *Information And Control*, vol. 8, pp. 338–353, 1965.

[13] B. Kosko, *Fuzzy Thinking: The New Science Of Fuzzy Logic*. Flamingo, 1993.