A Formal Approach to Identify Emergent Properties in Simulation Models

Yong Meng TEO*
Department of Computer Science
National University of Singapore (NUS)
email: teoym@comp.nus.edu.sg
url: www.comp.nus.edu.sg/~teoym

Acknowledgement:
Luong Ba Linh (NUS), Claudia Szabo
(University of Adelaide)

* Visiting Professor, Shanghai Advanced Research Institute,
Chinese Academy of Sciences, China
What is Emergence?

“The whole is greater than the sum of its parts”
- Aristotle
Complex Systems

- Complexity of systems is increasing
  - Large number of components
  - No global control, simple local rules
  - Non-linear connectivity among components

- Some system properties (emergent properties), do not result from the properties of its interconnected components

→ Component-based system with emergent properties
Component-based Systems

• Composed of multiple interacting components

• Component
  – “… a stand-alone functional element that is defined by its input and output behavior” [Hinton, 1997]

• Behavior
  – “A sequence of state changes it (component) undergoes during a period of time” [Chen, 2009]

• Property
  – Characteristic that can be detected at a point in time
Outline

• What is Emergence?

• Motivation

• Emergence Perspectives

• Formalizing Emergent Properties

• Propose Grammar-based Approach

• Example: Bird Flocking (extended Boids Model)

• Conclusion and Future Work
Examples/Applications of Emergence

• Game of life [Gardner, 1970]

• Flock of birds [Reynolds, 1987]

• Traffic network [Nagel, 1992]

• World Wide Web [Adamic, 2000]

• Stock market [Chen, 2002]

• More examples [Mogul, 2006]
Game of Life [Gardner, 1970]

• Micro level
  – Local knowledge
    • Each cell knows states of its eight neighbors
  – Simple behavior rules
    • A live cell with fewer than two live neighbors or more than three live neighbors becomes dead
    • A dead cell with exactly three live neighbors becomes alive

• Macro level
  – Some patterns of cells have the ability to move (glider) or reproduce (oscillators)

• Related studies of emergence in the Game of Life
  – [Chan, 2011], [Sanders, 2009], [Kubik, 2003], [Wuensche, 1999]
Traffic Network [Nagel, 1992]

• Micro level
  – Local knowledge
    • Each entity (vehicle, pedestrian, traffic light, etc.) has limited knowledge of its neighborhood.
  – Simple behaviors rules
    • Each entity obeys a set of simple rules (e.g. a vehicle obeys movement rules to avoid collision and maximize its speed)

• Macro level
  – Traffic congestion, traffic jam

• Related studies of emergence in traffic network
  – [Han, 2012], [Manley, 2010]
World Wide Web [Adamic, 2000]

• Micro level
  – **Local knowledge**
    • Each page links to some other pages
    • No central organization rationing the number of links
  – **Simple behavior rules**
    • No specific rules, i.e. a page can have a hyperlink pointing to any other page

• Macro level
  – The distribution of links follows a power law in which a few pages are linked to many times and most pages are seldom linked to
Stock Market

• Micro level
  – **Local knowledge**
    • Each investor has knowledge of a limited number of companies
    • No entity that controls the entire market
  – **Simple behavior rules**
    • Investors follow the regulatory rules of the market
    • Each investor tries to obtain the maximum possible profit

• Macro level
  – Relative security prices of companies across the world are regulated

• Related studies of emergence in stock market
  – [Chen, 2002]
Impacts of Emergence

• **Beneficial**
  – Positive additions to designed properties
    • Users adapt products to support tasks that designers never intended -> more competitive products
  – **Reduce the complexity** of a system [Chen, 2009]
    • Exploits self-organization

• **Harmful**
  – Leads to unforeseeable failures
  – Make a system harder to design, analyze, and control
Motivation

System Properties

Design Properties
- Syntactic Verification
- Semantic Validation

Emergent Properties
- Identify, Validation and Reason

2 proposed approaches:
- objective-based
- grammar-based (this talk)
Modeling & Simulation Lifecycle
- Design Properties

Our Work

- Component-based modeling
  - Component abstractions: meta-components, domain knowledge representation [ANSS 2008]

- Syntactic composability verification
  - EBNF composition grammar [AMS 2007, PADS2010]

- Semantic composability validation:
  - Approach [PADS 2009, PADS 2011]
  - General model properties [PADS2009, PADS2010]
  - Formal time-based formalism of model execution [Wintersim 2009*]
  - Costs of validation [PADS2011, JoS2012]

* ACM SIGSIM Best PhD Student Paper Award

Traditional Modeling & Simulation Lifecycle (adapted from Banks et al., 2005)

Proposed Component-based Modeling & Simulation Life-cycle [Teo2008]
Design Properties – Composability and Semantic Validation


29 January 2013 ISMS2013 - Keynote on Emergent Properties
Emergent Properties Detection & Validation


Emergence Perspectives

• **Philosophy**
  – Limitations of our knowledge
  – “The **whole** is **greater** than the **sum** of its parts.” [Aristotle]

• **Natural sciences, social sciences**
  – Explain emergence via theories in physics, biology, chemistry, and human behavior
  – Emergence is the formation of **order** from **disorder** based on **self-organization**. [Fromm, 2007], [Trianni, 2011]
  – A property is emergent if it is **unexplainable** from the properties of components at the **lower levels**. [Baas, 1997], [Johnson, 2008]
## Emergence Perspectives

<table>
<thead>
<tr>
<th>Perspective</th>
<th>Main Problem</th>
<th>Types of Emergence Studied</th>
<th>Concepts of Emergence</th>
<th>Examples</th>
<th>Issues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Philosophy</td>
<td>How intelligence emerges from unintelligent matter?</td>
<td>Strong emergence</td>
<td>More than the sum</td>
<td>Consciousness</td>
<td>Observer-dependent</td>
</tr>
<tr>
<td>Natural Sciences</td>
<td><strong>Biology</strong>: How life emerges from inanimate matter?</td>
<td>Simple emergence</td>
<td>Limitations in our knowledge</td>
<td>Bird flocking, Ant colonies</td>
<td>Mainly based on self-organization</td>
</tr>
<tr>
<td></td>
<td><strong>Chemistry</strong>: How a new substance emerges from the other ones?</td>
<td></td>
<td>Self-organization, Hierarchy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Social Sciences</td>
<td>How human behaviors emerges from the interactions between them?</td>
<td>Simple emergence</td>
<td></td>
<td>Crowd behavior</td>
<td></td>
</tr>
<tr>
<td>Computer Science</td>
<td>Detect and validate emergent properties</td>
<td>Weak emergence</td>
<td>Upward &amp; downward causation</td>
<td>Link distribution in WWW, Priority inversion in O/S</td>
<td>Lack of complete formalism of emergence</td>
</tr>
<tr>
<td></td>
<td>Reason about emergence</td>
<td>Multiple emergence</td>
<td>Shifts in complexity</td>
<td></td>
<td>Emergence validation</td>
</tr>
</tbody>
</table>

29 January 2013  ISMS2013 - Keynote on Emergent Properties
Computer Science (Our) Perspective

• Emergence is **intrinsic** to the system, observer-independent
• Emergence can be **reasoned**

• A property is emergent if it is both generated (**upward causation**) and autonomous (**downward causation**) from the underlying components. [Bedau, 2003]

• **Causation** is the relationship between a cause and an effect
Our Downward Causation-based Emergence Classification

System Properties

Designed Properties

Emergent properties

DC: downward causation

No DC

Positive or negative DC

Positive and negative DC

Complex DC

Simple Emergence

Weak Emergence

Multiple Emergence

Strong Emergence

Natural & Social Sciences

Computer Science

Philosophy

Negative DC

Positive DC

Stable Weak Emergence

Unstable Weak Emergence

Evolutionary Emergence

...
Simple Emergence

• Micro level
  – State of an entity is independent of
    • States of other entities
    • State of the system
    • Environment

• Macro level
  – No downward causation
Weak Emergence

• Micro level
  – Direct interaction
    • component interactions lead to the formation of groups and clusters
    • Groups and clusters influence in turn the behavior of the components
  – Indirect interaction through environment
    • Components change the state of the system and the environment through their behaviors
    • Changes in the environment influence in turn the behavior of components

• Macro level
  – can be derived from the micro level properties but only by simulation
  – Positive or negative downward causation
Stable Weak Emergence

- **Negative** downward causation

- There is a **balance** between
  - On the one hand: exploration, diversity and randomness through “creative” upward causation
    - Diversity and exploration due to autonomy and unique contexts of components
  - On the other hand: exploitation, unity and order through “constraining” downward causation
    - Unity and exploitation impose a constraint on agents
Examples: Stable Weak Emergence

• World Wide Web
  – Mirror the huge diversity of the world population, but all web pages follow the unifying standards and constraints of the W3C and other consortiums

• Wikipedia
  – Rely on the diversity of its participants and contributors, but every participant uses the same simple editor and obeys the same easy rules

• Flocks of birds
  – Birds in a flock fly in different directions to avoid collisions, but at the same time they try to match the neighbors’ velocity and steer to the perceived centre of the flock
Unstable Weak Emergence

- A form of undesirable, negative emergence
- Positive downward causation
- Examples

**Inflation in Economics**

- High prices → increase prices → High wages
- High wages → Increase wages → High cost of living
- High cost of living → increase cost of living → ...

**Domination of Some Social Networks**

- Interesting social networks (e.g. Facebook) attract Users
- Users make them expand
- Other social networks decline

Micro properties → + → + → + → Macro properties

---

29 January 2013

ISMS2013 - Keynote on Emergent Properties
Multiple Emergence

• Both positive and negative downward causation

• Example: stock market
  – Short-range positive downward causation
    • Stocks rise $\rightarrow$ encourage investors to buy $\rightarrow$ price rises even more
  – Long-range negative downward causation
    • Buyers know that there must be finally a peak and prices will drop $\rightarrow$ deterring buyers
Strong Emergence

• **not deducible**, even in principle, from the properties of the underlying components

• Similar to the notion of strong emergence of Bedau [Bedau, 1997] and Chalmers [Chalmers, 2006]

• Example: life is a strong emergent property of genes, genetic code and nucleic/amino acids
Evolutionary Emergence

• Highly complex adaptive properties

• Entity may add to or delete from its governing rule set in response to either local or global influences

• E.g. biological entities
Research Objective

To develop a formal approach for automatic identification and validation of emergent properties in component-based systems.
FORMALIZING EMERGENT PROPERTIES

• Emergence Formalisms

• Basic Emergence [Kubik, 2003]

• Issues in Kubik’s Approach
Emergence Formalisms

• Variable-based [Fisch, 2010]
  – One or more variables describe emergent properties
  – The system run is recorded and relationships are deduced
  – Assumes prior knowledge about emergence

• Event-based [Chen, 2009]
  – Behavior as series of simple and complex events
  – Emergence defined as sequence of complex events
  – Assumes prior knowledge about emergence

• Grammar-based [Kubik, 2003]
  – \( L_{\text{WHOLE}} \): set of properties of the system as a whole
  – \( L_{\text{SUM}} \): set of properties obtained from the union of the parts
  – Basic Emergence \( (L_{\xi}) = L_{\text{whole}} - L_{\text{sum}} \)
  – How to obtain \( L_{\text{sum}}, L_{\text{whole}} \)?
# Emergence Formalisms - Comparison

<table>
<thead>
<tr>
<th>Approach</th>
<th>Emergence Definition</th>
<th>Reason about Emergence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable-based</td>
<td>a priori</td>
<td>difficult</td>
</tr>
<tr>
<td>[Fisch,2010],[Seth,2008],[Mnif,2006]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Event-based</td>
<td>a priori</td>
<td>possible</td>
</tr>
<tr>
<td>[Chen,2009]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grammar-based</td>
<td>not required</td>
<td>possible</td>
</tr>
<tr>
<td>[Kubik,2003]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Formal Definition of Basic Emergence [Kubik, 2003]

• Basic emergence – “behavior reducible to agent-to-agent interactions without any evolutionary processes involved”

• Defines basic emergence using grammars

• Grammars:
  – alphabet \((V) = \{\text{symbols}\}\)
    • Power of alphabet \((V^k) = \text{set of strings of length } k\)
  – word\((w) = \text{a reachable string of symbols}\)
  – rewriting rules \((R)\) translate a word to another word
Model of a System

- A multi-agent system consists of agents (A) interacting in an environment (E) that is subdivided into cells (e)

- Behavior (L) of an agent is characterized by rule-based behavioral descriptions (behavior rules R)

- States:
  - System state (S)
  - State of the environment (S_E) and cell e (s_e)
  - State of all agents (S_A) and agent instance A_j (s_j)
Grammar-based System Model

• A system of \( n \) agents, \( A_1, A_2, \ldots, A_n \), interacting in an environment (E) consisting of \( c \) cells is defined as

\[
\text{System} = (V_A, V_E, A_1, A_2, \ldots, A_n, S(0))
\]

where

- \( V_A \): set of possible agent states
- \( V_E \): set of possible cell states
- \( V = V_A \cup V_E \)
- \( A_j \): agent instance \( j \) \((1 \leq j \leq n)\)
- \( S(t) \in V^{c+n} \): system state at time \( t \), and \( S(0) \) denotes the initial system state at time 0.
Environment

• Cell (e)
  – $V_e$: set of possible states of cell $e$
  – $s_e(t) \in V_e$: state at time $t$, and $s_e(0)$ denotes the initial state at time 0

• Environment (E)
  – $V_E = \bigcup_{e=1}^{c} V_e$
  – $S_E(t) \in V_E^c$: state at time $t$, and $S_E(0)$ denotes the initial state at time 0
Agents

• Agent $A_j$ is defined as
  
  \[ A_j = (V_j, R_j, s_j(0)) \]

  where
  
  – $V_j \subseteq V_A$: set of possible states of $A_j$
  – $R_j$: set of behavior rules for $A_j$
  – $s_j(t) \in V_j$: state of $A_j$ at time $t$, and $s_j(0)$ denotes the initial state of $A_j$ at time $0$

• Behavior rules of $A_j$ ($R_j$):
  
  \[ R_j: s_j(t) \rightarrow s_j(t+1) \text{ or } R_j: V_j \rightarrow V_j \]
Issues in Kubik’s Approach

• Broad definition of emergence

• Emergent formalism:
  1. agent types
     [ introduce agent type, $A_{ij}$ type $i$ ($1 \leq i \leq m$) ]
  2. mobile agents
     [ define mobility as attributes of agents:
       $P_i = P_{i\text{mobile}} \cup P_{i\text{others}}$ ]
  3. Open system
     Game of life example, agents exist throughout the observation period, and number of agents ($n$) = number of cells ($c$)
     [ agents can enter and leave system – model open system, $n$ varies over time ]
PROPOSED GRAMMAR-BASED APPROACH

• Objectives

• Overview of Approach

• System Model

• Proposed Grammar

• Example: Bird Flocking (extended Boids Model)
Proposed Emergence Definition

Kubik: \( L_\xi = L_{\text{whole}} - L_{\text{sum}} \)

Proposed:
\[
egin{align*}
L_{\text{whole}} &= L_{\text{whole}}^{\text{NI}} + L_{\text{whole}}^I \\
L_{\text{sum}} &= L_{\text{sum}}^{\text{NP}} + L_{\text{sum}}^P \\
L_\xi &= L_{\text{whole}}^I - L_{\text{sum}}^P
\end{align*}
\]
Design Considerations

• Computer science perspective

• Basic (weak) emergence

• Component-based systems - modeled as a multi-agent system

• Extended grammar-based formalism: agent types, mobile agents, open systems

• Automatic detection, validation and reasoning of emergence
Overview of Approach

Component-based System

System Model

Agent

Agent

Proposed Grammar-based System Model

agent-based modeling

grammar-based formalism

Design Properties

Emergent Properties

Detection, Validation and Reason
System Model

• A system is modeled as a multi-agent system consisting of different types of agents (A) interacting in an environment (E). E is modeled as a 2D grid that is subdivided into units called cells (e)

• **Behavior (L)** of an agent is characterized by rule-based behavioral descriptions (behavior rules (R))

• **States:**
  – System state (S)
  – State of the environment (S_E) and cell e (s_e)
  – State of all agents (S_A) and agent A_{ij} (s_{ij})
Proposed Grammar

A system consisting of \( m \) agent types and \( n \) agents \( A_{1n_1}, \ldots, A_{mn_m} \) (\( n = n_1 + \ldots + n_m \), \( n_i \) agents of type \( i \)) interacting in an environment (2D grid) consisting of \( c \) cells is defined as

\[
\text{GBS} = (V_A, V_E, A_{1n_1}, \ldots, A_{mn_m}, S(0))
\]

where

- \( A_{ij} \): agent instance \( j \) (\( 1 \leq j \leq n_i \)) of type \( i \) (\( 1 \leq i \leq m \))
- \( V_A = \bigcup_{i=1}^{m} V_{A_i} \): set of possible agent states for all agent types, and \( V_{A_i} \) denotes the set of possible states for agents of type \( i \)
- \( V_E \): set of possible cell states
- \( V = V_A \cup V_E \)
- \( S(t) \in V^{c+n} \): system state at time \( t \), and \( S(0) \) denotes the initial system state at time 0
Agents

- Agent $A_{ij} \ (1 \leq i \leq m, \ 1 \leq j \leq n)$, is defined as
  \[ A_{ij} = (P_i, R_i, s_{ij}(0)) \]

where
- $P_i$: set of attributes for agents of type $i$
  \[ P_i = P_{i\_mobile} \cup P_{i\_others} \]
  \[ P_{i\_mobile} = \{x|x \text{ is an attribute that models mobility}\} \]
  \[ P_{i\_others} = P_i \setminus P_{i\_mobile} \]

- $V_{A_i}$: set of possible states for agents of type $i$

- $R_i$: set of behavior rules for agents of type $i$
  \[ R_i = R_{i\_mobile} \cup R_{i\_others} \]
  \[ R_{i\_mobile}: V_{A_i} \rightarrow V_{A_i} \]
  \[ R_{i\_others} = P_i \setminus P_{i\_mobile} \]

- $s_{ij}(t) \in V_{A_i}$: state of $A_{ij}$ at time $t$, and $s_{ij}(0)$ denotes the initial state of $A_{ij}$ at time 0
\[ L^I_{\text{whole}} \]

- Behavior of the system as a whole

\[ L^I_{\text{whole}} = \{ w \in V^{c+n} | S(0) \Rightarrow^*_{\text{GROUP}} w \} \]

- Includes interactions among agents (GROUP)

- \( L^I_{\text{whole}} \) returns a set of words (w) that represents the set of system states reachable from an initial system state, S(0)
\( L^P_{\text{sum}} \)

- Sum of agent behaviors:
  \[ L^P_{\text{sum}} = \text{superimpose}(L(A_{1n1}),..., L(A_{mnm})) \]

returns a set of words resulting from superimposing words of behaviors of individual agents

- \( L(A_{ij}) \) - behavior of an agent \( A_{ij} \), is defined as
  \[ L(A_{ij}) = \{ w \in V^{c+1} \mid (s_{ij}(0) \cup S_E(0)) \Rightarrow^* w \} \]

considering only this agent in the system, behavior returns a set of words (\( w \)) that represents the set of system states reachable from an initial system state, \( s_{ij}(0) \cup S_E(0) \)
Example: Bird Flocking

• Boids model [Reynolds, 1987] captures the motion of bird flocking and is a seminar example for studying emergence [Szabo, 2012], [Chan, 2011]

• Extension of the boids model with two types of birds, ducks and geese
Agent-based Model

• The system is modeled as a multi-agent system consisting of two types of agents (duck and goose) interacting in an environment represented as a 2D grid (E) that is subdivided into cells (e) of equal size.

• Behavior (L) of an agent is characterized by rule-based behavioral descriptions (behavior rules (R))

• States:
  – System state (S)
  – State of the environment (SE) and cell e (s_e)
  – State of all agents (SA) and agent A_ij (s_ij)
Environment

- 2D grid of 8x8 cells

- Cell states
  - occupied (by a bird)
  - free (not occupied)

- A cell fits one duck or one goose

- Neighborhood distance: 3 cells
Agents – Ducks and Geese

• Attributes
  – Color: duck – red, goose - blue (for visualization)
  – Mobility:
    • Position: position of agent in the environment
    • Velocity:
      – speed: 0, 1, or 2 cells per step
      – 8 directions: north, north-east, east, south-east, south, south-west, west, north-west
      – represented as a vector (discrete)

• Behavior Rules [Reynolds 1987]
  – Separation: avoid collision with nearby birds
  – Alignment: fly as fast as nearby of the same type
  – Cohesion: stay close to nearby of the same type
Proposed Grammar-based Formalism

\[ \text{GBS} = (V_A, V_E, A_{11}, \ldots, A_{15}, A_{21}, \ldots, A_{25}, S(0)) \]

where

- Agent types: 1 denotes duck, 2 denotes goose
- \( A_{1j} \) is duck instance \( j \) (\( 1 \leq j \leq 5 \)), and \( A_{2j} \) is goose instance \( j \) (\( 1 \leq j \leq 5 \))
- \( V_A = V_{A1} \cup V_{A2} \): set of possible states for the ducks \( (V_{A1}) \) and the geese \( (V_{A2}) \)
- \( V_E \): set of possible cell states
- \( V = V_A \cup V_E \)
- \( S(t) \in V^{64+10} \): system state at time \( t \), and \( S(0) \) denotes the initial system state at time 0
Environment

• Cell (e)
  – \( V_e = \{o, f\} \), where occupied (o), free (f)
  – \( s_e(t) \in V_e \)

• Environment (E)
  – \( V_E = \bigcup_{e=1}^{64} V_e = \{o, f\} \)
  – \( S_E(t) \in V_E^{64} \)
Agents – Ducks

- Duck instance $A_{1j}$ ($1 \leq j \leq 5$) is defined as
  $$A_{1j} = (P_1, R_1, s_{1j}(0))$$
  where
  - $P_1 = P_{1\_mobile} \cup P_{1\_others}$
    $$P_{1\_mobile} = \{\text{position}(g_{1j}), \text{velocity}(v_{1j})\}, \quad P_{1\_others} = \{\text{color}\}$$
  - $V_{A_1} = \{(x,y) | 1 \leq x \leq 8; 1 \leq y \leq 8\} \times \{\{(\alpha,\beta) | -2 \leq m \leq 2; -2 \leq n \leq 2\} \times \{\text{red}\}\}$
  - $R_1 = R_{1\_mobile} \cup R_{1\_others}$
    $$R_{1\_mobile}, \quad R_{1\_others} = \emptyset$$
  - $s_{1j}(t) \in V_{A_1}$

Similarly for Geese!
Behavior Rules for Ducks – $R_{1\_mobile}^1$

- For duck instance $A_{1j}$ ($1 \leq j \leq 5$) at time $t$ with position $g_{1j}(t)$ and velocity $v_{1j}(t)$:

  - **Update position:**
    $$g_{1j}(t+1) = g_{1j}(t) + v_{1j}(t+1)$$

  - **Update velocity:**
    $$v_{1j}(t+1) = v_{1j}(t) + \text{separation}(A_{1j}) + \text{alignment}(A_{1j}) + \text{cohesion}(A_{1j})$$

Similarly for Geese!
Separation Rule

• Goal: avoid collision with nearby *birds*

• How: if *duck a* is close to another *bird b*, i.e. within $\varepsilon$ cells, then *a* flies away from *b*

\[
\text{separation}(a) = \sum_{\text{distance}(a,b) \leq \varepsilon} a.\text{position} - b.\text{position}
\]

```
separation(duck a)
vector c = 0;
for each boid b
    if |a.position - b.position| \leq \varepsilon then
        c = c - (b.position - a.position)
return c
```
Alignment Rule

- Goal: fly as fast as nearby ducks

- How: change velocity of duck a by $\lambda\%$ towards the average velocity of its neighboring ducks

\[
\text{alignment}(a) = \left( \sum_{\text{duck}(b) \in \text{neighbor}(a,b)} b.\text{velocity} \right) / k - a.\text{velocity} / \lambda
\]

alignment(duck a)
vector c = 0;
integer k = 0;
for each neighboring duck b
  k = k + 1;
  c = c + b.velocity;
endfor
  c = c / k;
return (c - a.velocity) / $\lambda$
Cohesion Rule

- **Goal:** stay close to nearby *ducks*

- **How:** move *duck a* by $\gamma\%$ towards the center of its neighboring ducks

\[
\text{cohesion}(a) = \left( \sum_{\text{duck}(b) \in \text{neighbor}(a,b)} b.\text{position} \right) / k - a.\text{position} / \gamma
\]

```plaintext
cohesion(duck a)
    vector c = 0;
    integer k = 0;
    for each neighboring duck b
        k = k + 1;
        c = c + b.\text{position};
    endfor
    c = c / k;
return (c - a.\text{position}) / \gamma
```
Example of Emergence Detection

• Objective: show how flocking of birds, a known emergence, is detected

• Flocking - at least 4 birds of the same type fly together

• Together - each bird has at least one immediate neighbor of the same type
Example of Emergence Detection

- Given an initial system state:

<table>
<thead>
<tr>
<th></th>
<th>1, (1,-1)</th>
<th>2, (1,0)</th>
<th></th>
<th>3, (1,0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(1,0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>(1,0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>(1,0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(1,0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(1,0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>(1,1)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

where red denotes duck and blue denotes goose, <j, (α,β)> denotes a bird, with velocity (α,β)

- Determine basic emergence = \( L_{\text{whole}}^I - L_{\text{sum}} \)
## Vector Representation of Velocity

<table>
<thead>
<tr>
<th>Direction</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td>North</td>
<td>(0,0)</td>
</tr>
<tr>
<td>North-east</td>
<td>(0,0)</td>
</tr>
<tr>
<td>East</td>
<td>(0,0)</td>
</tr>
<tr>
<td>South-east</td>
<td>(0,0)</td>
</tr>
<tr>
<td>South</td>
<td>(0,0)</td>
</tr>
<tr>
<td>South-west</td>
<td>(0,0)</td>
</tr>
<tr>
<td>West</td>
<td>(0,0)</td>
</tr>
<tr>
<td>North-west</td>
<td>(0,0)</td>
</tr>
</tbody>
</table>

29 January 2013

ISMS2013 - Keynote on Emergent Properties
System states at $t=4$ and $5$ are different but emergence property [bird flocking pattern and velocity] is similar.
\[ L^I_{\text{whole}} \]

• Observation
  – At \( t=4 \) and \( t=5 \), the system states are different but the emergent property is the same [bird flocking pattern and velocity]
  – \( S(12) = S(4) \)

• Hence by definition,

\[ L^I_{\text{whole}} = \{ S(0), S(1), S(2), S(3), S(4), S(5), ..., S(11) \} \]
\[ L_{\text{sum}} \]

- \( L_{\text{sum}} = \text{superimpose}(L(A_{11}), \ldots, L(A_{15}), L(A_{21}), \ldots, L(A_{25})) \)

- For illustration, consider two geese:
  \[ L_{\text{sum}} = \text{superimpose}(L(A_{23}), L(A_{25})) \]
\( L(A_{23}) \) and \( L(A_{25}) \)

\[
L(A_{23}) = \begin{cases} 
    \{s_{23}(0), s_{23}(1), \ldots, s_{23}(7)\} \text{ where } s_{23}(0) = \{f, f, \ldots, (\text{blue, (1,2), (1,0)), f, f, \ldots, f)\} 
\end{cases}
\]

\[
L(A_{25}) = \begin{cases} 
    \{s_{25}(0), s_{25}(1), \ldots, s_{25}(7)\} 
\end{cases}
\]

- **L(A_{23})**
  - At time \( t=0 \):
  - Fly east with speed of 1 cell.
  - \( s_{23}(0) = \{f, f, \ldots, (\text{blue, (1,2), (1,0)), f, f, \ldots, f)\} \)
  - For each subsequent time step, the fly moves one cell to the east.

- **L(A_{25})**
  - At time \( t=0 \):
  - Fly north-east with speed of 1 cell.
  - \( s_{25}(0) = \{f, f, \ldots, (\text{blue, (1,2), (1,0)), f, f, \ldots, f)\} \)
  - For each subsequent time step, the fly moves one cell to the north-east.
\[ L_{\text{sum}} = \text{superimpose}(L(A_{23}), L(A_{25})) \]

\[ = L(A_{23}) \text{ superimpose } (L(A_{25})) \cup L(A_{25}) \text{ superimpose } (L(A_{23})) \]

\[
\begin{align*}
L(A_{23}) \text{ superimpose } (L(A_{25})) &= \{(f, f, f \ldots (\text{blue}, (1,2), (1,0)), \ldots (\text{blue}, (1,1), (1,1)), \ldots), \\
&\quad (f, f, f \ldots (\text{blue}, (2,2), (1,0)), \ldots), \\
&\quad (f, f, f \ldots (\text{blue}, (3,3), (1,1)), \ldots (\text{blue}, (3,2), (1,0)), \ldots), \\
&\quad \ldots \} 
\end{align*}
\]
Emergent Property States – Flocks of birds

• \( L_\xi = \{S(1), S(2), S(3), S(4), \ldots, S(11)\} \)

• Based on flocking definition (at least 4 birds of the same type fly together):
  1. known emergent states - \( S(2), S(3), S(4), \ldots, S(11) \)
  2. unknown emergent state - \( S(1) \)
Conclusion

• Grammar-based approach to identify and validate emergent properties
  – Tighter definition of emergence
  – Emergent formalism: agent types, mobile agents, open systems

• Extended boids model
  – Large problem size can lead to state explosion problem
  – Mobile agents have more impacts on complexity
Further Work

• How to determine $L^p_{\text{sum}}$?

• Defining known emergence and making sense of unknown emergence

• Reasoning of emergence – useful, harmful, ...

• Applications – road traffic networks, concurrent program verification, social networks, ...
Thank you

http://www.comp.nus.edu.sg/~teoym
Email: teoym@comp.nus.edu.sg
References


References