Modelling and analysis of collective adaptive systems

Michele Loreti

UNIVERSITÀ DEGLI STUDI FIRENZE

quanticol

Based on joint work with
Yehia Abd Alrahman, Rocco De Nicola, Jane Hillston

Annual Meeting of IFIP Working Group 2.2
Singapore, 12-16 September 2016

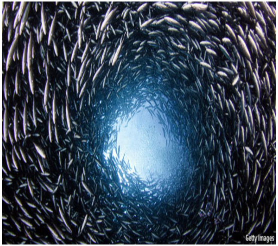
Collective Systems

We are surrounded by examples of collective systems:

Collective Systems

We are surrounded by examples of collective systems:

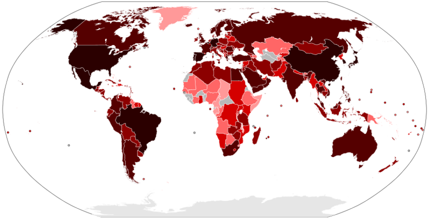
in the natural world ....

Collective Systems

We are surrounded by examples of collective systems:

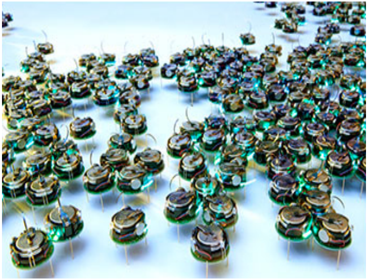
.... and in the man-made world



Collective Systems

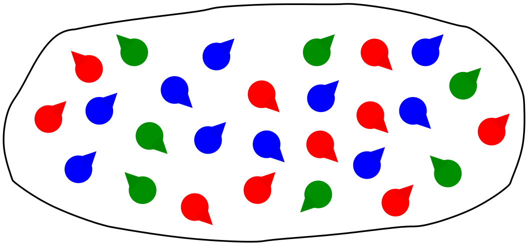
We are surrounded by examples of collective systems:

.... and in the man-made world



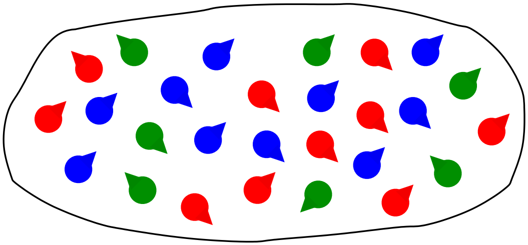
Collective Adaptive Systems

From a computer science perspective these systems can be viewed as being made up of a large number of interacting entities.



Collective Adaptive Systems

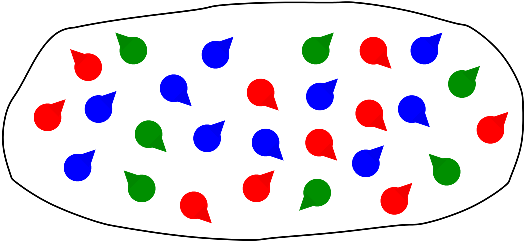
From a computer science perspective these systems can be viewed as being made up of a large number of interacting entities.



Each entity may have its own properties, objectives and actions.

Collective Adaptive Systems

From a computer science perspective these systems can be viewed as being made up of a large number of interacting entities.

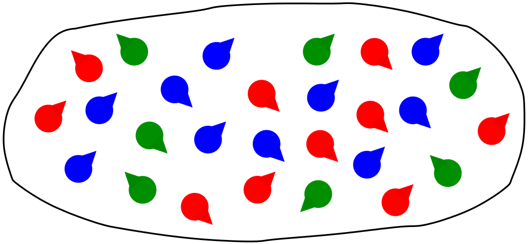


Each entity may have its own properties, objectives and actions.

At the system level these combine to create the collective behaviour.

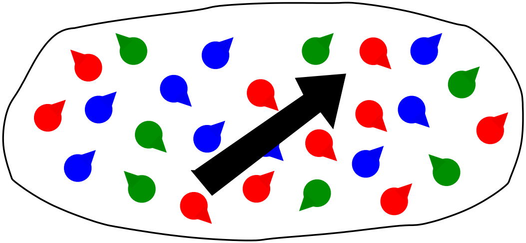
Collective Adaptive Systems

The behaviour of the system is thus dependent on the behaviour of the individual entities.



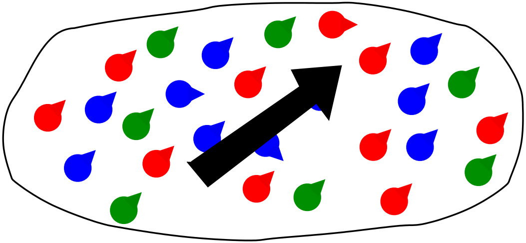
Collective Adaptive Systems

The behaviour of the system is thus dependent on the behaviour of the individual entities.



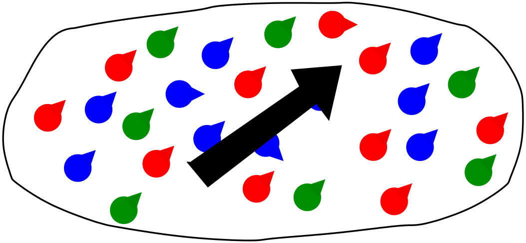
Collective Adaptive Systems

The behaviour of the system is thus dependent on the behaviour of the individual entities.



Collective Adaptive Systems

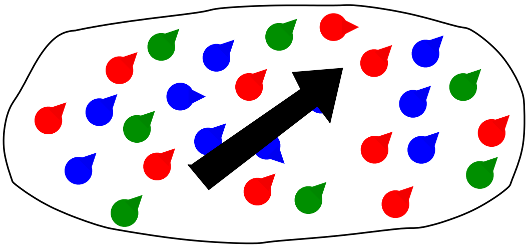
The behaviour of the system is thus dependent on the behaviour of the individual entities.



And the behaviour of the individuals will be influenced by the state of the overall system.

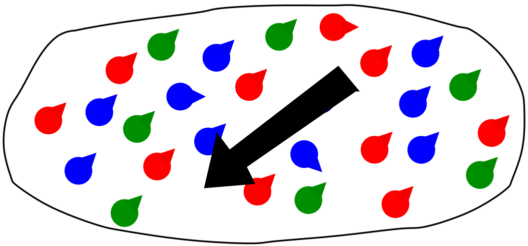
Collective Adaptive Systems

CAS are often embedded in our environment and need to operate without centralised control or direction.



Collective Adaptive Systems

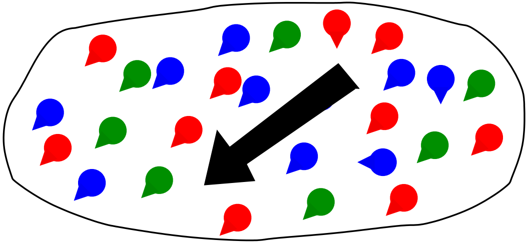
CAS are often embedded in our environment and need to operate without centralised control or direction.



Moreover when conditions within the system change it may not be feasible to have human intervention to adjust behaviour appropriately.

Collective Adaptive Systems

CAS are often embedded in our environment and need to operate without centralised control or direction.



Moreover when conditions within the system change it may not be feasible to have human intervention to adjust behaviour appropriately.

Thus systems must be able to autonomously adapt.

Challenges for modelling CAS

Works on Process Algebra provide a solid basic framework for modelling CAS but there remain a number of challenges:

Challenges for modelling CAS

Works on Process Algebra provide a solid basic framework for modelling CAS but there remain a number of challenges:

- Richer forms of interaction

Challenges for modelling CAS

Works on Process Algebra provide a solid basic framework for modelling CAS but there remain a number of challenges:

- Richer forms of interaction
- Capturing adaptivity

Challenges for modelling CAS

Works on Process Algebra provide a solid basic framework for modelling CAS but there remain a number of challenges:

- Richer forms of interaction
- Capturing adaptivity
- The influence of *environment* on behaviour

Modelling and analysis of CAS

Our goal is to develop a coherent, integrated set of linguistic primitives, methods and tools to build systems that can operate in open-ended, unpredictable environments.

Modelling and analysis of CAS

Our goal is to develop a coherent, integrated set of linguistic primitives, methods and tools to build systems that can operate in open-ended, unpredictable environments.

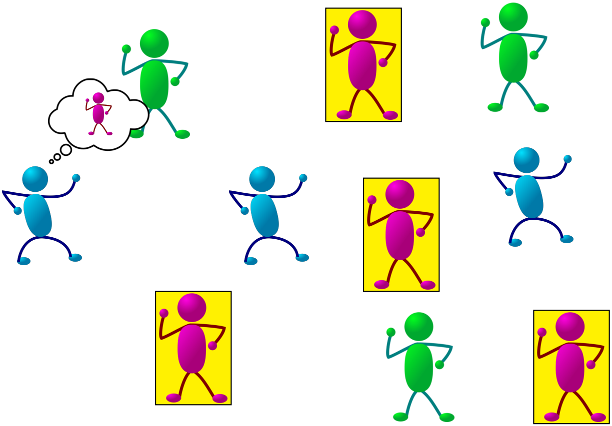
In this talk:

1. *attribute based communication* for modelling interactions in CAS;
2. CARMA: a language for modelling CAS;
3. a simple example.

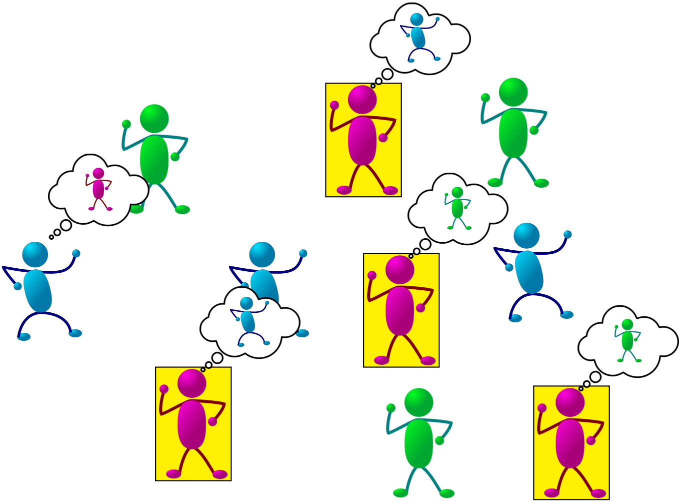
Attribute-based communication

Attribute-based communication

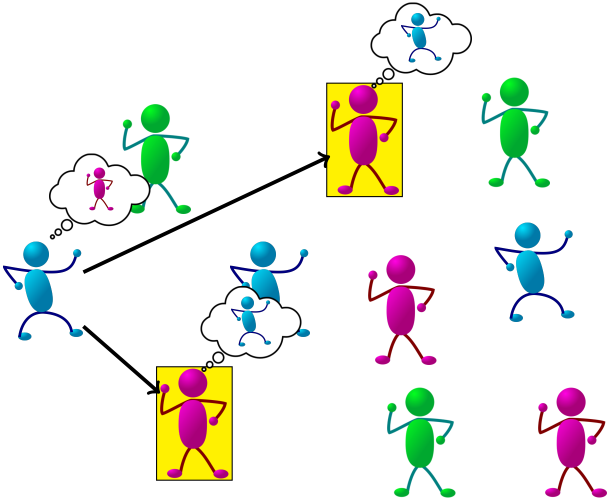
Attribute-based communication



Attribute-based communication



Attribute-based communication



Attribute-based communication

Abd Alrahman et al.[1] presented a general theory for *attribute based communication*:

- *AbC*, a calculus for attribute-based communication;
- encoding of classical *interaction patterns* via *attribute based* communications;
- a behavioural theory for AbC.

[1] Yehia Abd Alrahman, Rocco De Nicola, and Michele Loreti. "On the Power of Attribute-Based Communication". In: *FORTE 2016, Proceedings*. Ed. by Elvira Albert and Ivan Lanese. Vol. 9688. Lecture Notes in Computer Science. Springer, 2016, pp. 1–18.

Attribute-based communication

Abd Alrahman et al.[1] presented a general theory for *attribute based communication*:

- *AbC*, a calculus for attribute-based communication;
- encoding of classical *interaction patterns* via *attribute based* communications;
- a behavioural theory for AbC.

This work is focussed on *qualitative aspects* of CAS. In this talk we will focus on *quantitative aspects*.

[1] Yehia Abd Alrahman, Rocco De Nicola, and Michele Loreti. "On the Power of Attribute-Based Communication". In: *FORTE 2016, Proceedings*. Ed. by Elvira Albert and Ivan Lanese. Vol. 9688. Lecture Notes in Computer Science. Springer, 2016, pp. 1–18.

Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

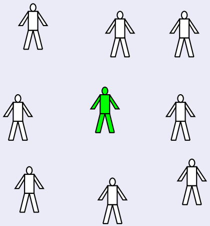
1. **Spreading**: one agent spreads relevant information to a given group of other agents

Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents

Spreading: 1-to-many

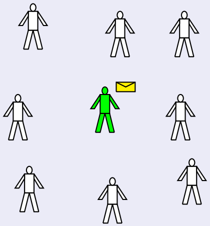


Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents

Spreading: 1-to-many

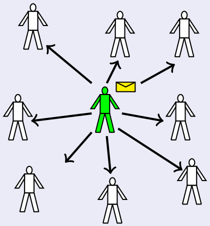


Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents

Spreading: 1-to-many

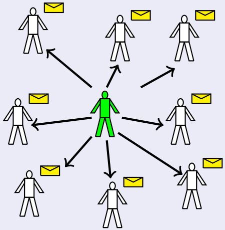


Interaction patterns in CAS

Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents

Spreading: 1-to-many

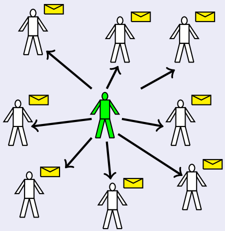


Interaction patterns in CAS

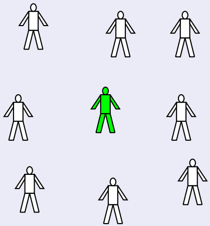
Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents
2. **Collecting**: one agent changes its behaviour according to data collected from one agent belonging to a given group of agents.

Spreading: 1-to-many



Collecting: 1-to-1

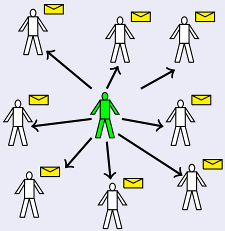


Interaction patterns in CAS

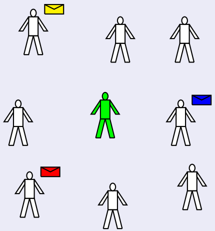
Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents
2. **Collecting**: one agent changes its behaviour according to data collected from one agent belonging to a given group of agents.

Spreading: 1-to-many



Collecting: 1-to-1

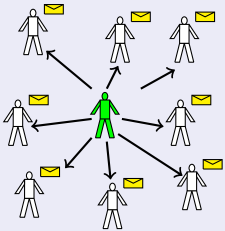


Interaction patterns in CAS

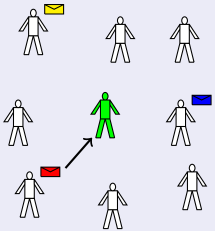
Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents
2. **Collecting**: one agent changes its behaviour according to data collected from one agent belonging to a given group of agents.

Spreading: 1-to-many



Collecting: 1-to-1

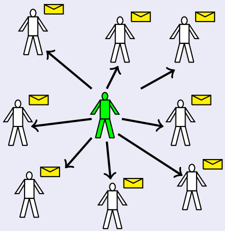


Interaction patterns in CAS

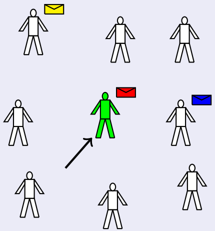
Typically, CAS exhibit two kinds of interaction pattern:

1. **Spreading**: one agent spreads relevant information to a given group of other agents
2. **Collecting**: one agent changes its behaviour according to data collected from one agent belonging to a given group of agents.

Spreading: 1-to-many



Collecting: 1-to-1



Interaction primitives for CAS

The following interaction primitives, all based on *attribute oriented communication*, can be considered for CAS:

Interaction primitives for CAS

The following interaction primitives, all based on *attribute oriented communication*, can be considered for CAS:

- **Broadcast output**: a message is sent to all the components satisfying a predicate π;
- **Broadcast input**: a process is willing to receive a broadcast message from a component satisfying a predicate π;

Interaction primitives for CAS

The following interaction primitives, all based on *attribute oriented communication*, can be considered for CAS:

- **Broadcast output**: a message is sent to all the components satisfying a predicate π;
- **Broadcast input**: a process is willing to receive a broadcast message from a component satisfying a predicate π;
- **Unicast output**: a message is sent to one of the components satisfying a predicate π;
- **Unicast input**: a process is willing to receive a message from a component satisfying a predicate π.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

1. The behaviours of agents and their interactions;

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

1. The behaviours of agents and their interactions;
2. The global knowledge of the system and that of its agents;

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

1. The behaviours of agents and their interactions;
2. The global knowledge of the system and that of its agents;
3. The environment where agents operate...

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

Carma: a process specification language for CAS

To support design of CAS we introduced Carma[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

1. The behaviours of agents and their interactions;
2. The global knowledge of the system and that of its agents;
3. The environment where agents operate...
   - taking into account open ended-ness and adaptation;

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

1. The behaviours of agents and their interactions;
2. The global knowledge of the system and that of its agents;
3. The environment where agents operate...
   - taking into account open ended-ness and adaptation;
   - taking into account resources, locations and visibility/reachability issues.

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

CARMA: a process specification language for CAS

To support design of CAS we introduced CARMA[2] (*Collective Adaptive Resource-sharing Markovian Agents*). This is a process specification language which handles:

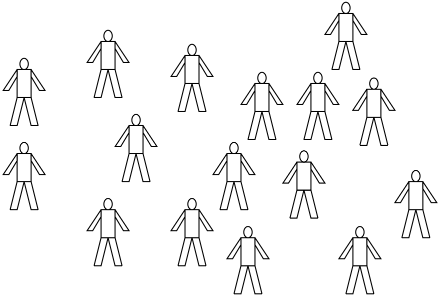
1. The behaviours of agents and their interactions;
2. The global knowledge of the system and that of its agents;
3. The environment where agents operate...
   - taking into account open ended-ness and adaptation;
   - taking into account resources, locations and visibility/reachability issues.

In CARMA the execution of an action takes an exponentially distributed time; the rate of each action is determined by the environment.

[2] Michele Loreti and Jane Hillston. "Modelling and Analysis of Collective Adaptive Systems with CARMA and its Tools". In: *SFM 2016*. Vol. 9700. Lecture Notes in Computer Science. Springer, 2016, pp. 83–119.

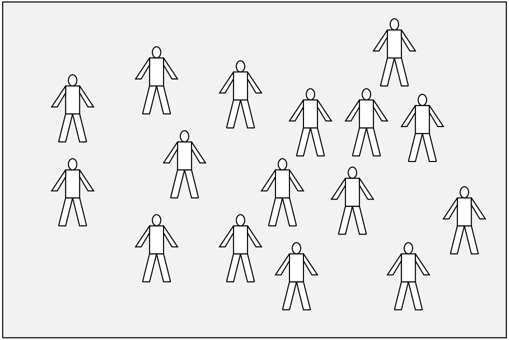
CAS: CARMA perspective

Collective



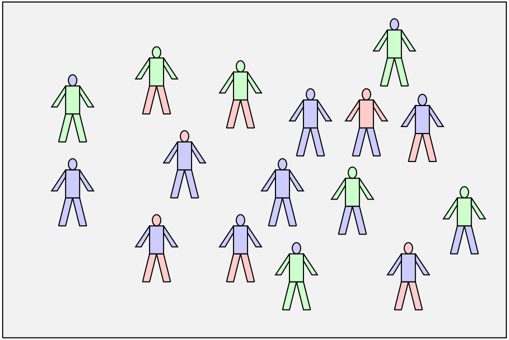
CAS: CARMA perspective

Collective **Environment**



CAS: CARMA perspective

Collective **Environment** **Attributes**



CARMA

A CARMA system consists of

CARMA

A CARMA system consists of

- a collective (N)...

CARMA

A CARMA system consists of

- a collective (N)...
- ...operating in an environment ($\mathscr{E}$).

CARMA

A CARMA system consists of

- a collective (N)...
- ...operating in an environment ($\mathscr{E}$).

Collective...

- is composed by a set of components, i.e. the Markovian agents that compete and/or cooperate to achieve a set of given tasks
- models the behavioural part of a system

CARMA

A CARMA system consists of

- a collective (N)...
- ...operating in an environment ($\mathscr{E}$).

Collective...

- is composed by a set of components, i.e. the Markovian agents that compete and/or cooperate to achieve a set of given tasks
- models the behavioural part of a system

Environment...

- models the rules intrinsic to the context where agents operate;
- mediates and regulates agent interactions.

Components

Agents in CARMA are defined as components C of the form (P, γ) where...

- P is a process, representing agent behaviour;
- γ is a store, modelling agent knowledge.

Components

Agents in CARMA are defined as components C of the form (P, γ) where...

- P is a process, representing agent behaviour;
- γ is a store, modelling agent knowledge.

Process Syntax:

$$P, Q ::= \textbf{nil} \mid act.P \mid P + Q \mid P \mid Q \mid [\pi]P \mid \textbf{kill} \mid A$$

Components

Agents in CARMA are defined as components C of the form (P, γ) where. . .

- P is a process, representing agent behaviour;
- γ is a store, modelling agent knowledge.

Process Syntax:

$$P, Q ::= \mathbf{nil} \mid act.P \mid P + Q \mid P \mid Q \mid [\pi]P \mid \mathbf{kill} \mid A$$

Store γ is a mapping from *attribute names* to *values*.

Interaction primitives

Syntax

$$
\begin{array}{rcll}
act & ::= & \alpha^{\star}[\pi]\langle\overrightarrow{e}\rangle\sigma & \text{Broadcast output} \\
 & \mid & \alpha^{\star}[\pi](\overrightarrow{x})\sigma & \text{Broadcast input} \\
 & \mid & \alpha[\pi]\langle\overrightarrow{e}\rangle\sigma & \text{Unicast output} \\
 & \mid & \alpha[\pi](\overrightarrow{x})\sigma & \text{Unicast input}
\end{array}
$$

- α is an action type;
- π is a predicate;
- σ is the effect of the action on the store.

Updating the store

After the execution of an action, a process can update the component store:

- σ denotes a function mapping each γ to a probability distribution over possible stores.

Updating the store

After the execution of an action, a process can update the component store:

- σ denotes a function mapping each γ to a probability distribution over possible stores.

$$\text{move}^{\star}[\pi]\langle v\rangle\{x := x + U(-1,+1)\}$$

Updating the store

After the execution of an action, a process can update the component store:

- σ denotes a function mapping each γ to a probability distribution over possible stores.

$$\text{move}^{\star}[\pi]\langle v\rangle\{x := x + U(-1,+1)\}$$

Remark:

- Processes running in the same component can implicitly interact via the local store;
- Updates are instantaneous.

More on synchronisation

Predicates regulating broadcast/unicast inputs can refer also to the received values.

More on synchronisation

Predicates regulating broadcast/unicast inputs can refer also to the received values.

Example:

A value greater than 0 is expected from a component with a *trust_level* less than 3:

$$\alpha^{\star}[(x > 0) \wedge (trust\_level < 3)](x)\sigma.P$$

More on synchronisation

Predicates regulating broadcast/unicast inputs can refer also to the received values.

Example:

A value greater than 0 is expected from a component with a *trust_level* less than 3:

$$\alpha^{\star}[(x > 0) \wedge (trust\_level < 3)](x)\sigma.P$$

Pattern matching can be encoded in CARMA.

Examples of interactions...

Broadcast synchronisation:

$$
\begin{array}{l}
(\ \mathrm{stop}^{\star}[\mathrm{bl} < 5\%]\langle v\rangle\sigma_1.P\ ,\{role = \text{``}master\text{''}\})\ \|\\
\quad (\ \mathrm{stop}^{\star}[\mathrm{role} = \text{``}master\text{''}](x)\sigma_2\ .Q_1\ ,\{\mathrm{bl} = 4\%\})\ \|\\
\qquad (\ \mathrm{stop}^{\star}[\mathrm{role} = \text{``}super\text{''}](x)\sigma_3.Q_2\ ,\{\mathrm{bl} = 2\%\})\ \|\\
\qquad\qquad (\ \mathrm{stop}^{\star}[\top](x)\sigma_4.Q_3\ ,\{\mathrm{bl} = 2\%\})
\end{array}
$$

Examples of interactions...

Broadcast synchronisation:

$$(\;\mathrm{stop}^{\star}[\mathrm{bl} < 5\%]\langle v\rangle\sigma_1.P \;,\{role = \text{“}master\text{”}\}) \parallel$$
$$(\;\mathrm{stop}^{\star}[\mathrm{role} = \text{“}master\text{”}](x)\sigma_2 \;.Q_1 \;,\{\mathrm{bl} = 4\%\}) \parallel$$
$$(\;\mathrm{stop}^{\star}[\mathrm{role} = \text{“}super\text{”}](x)\sigma_3.Q_2 \;,\{\mathrm{bl} = 2\%\}) \parallel$$
$$(\;\mathrm{stop}^{\star}[\top](x)\sigma_4.Q_3 \;,\{\mathrm{bl} = 2\%\})$$

Examples of interactions...

Broadcast synchronisation:

$$(\text{stop}^\star[\text{bl} < 5\%]\langle v\rangle\sigma_1.P \ ,\{role = \text{``}master\text{''}\}) \parallel$$
$$(\text{stop}^\star[\text{role} = \text{``}master\text{''}](x)\sigma_2 \ .Q_1 \ ,\{\text{bl} = 4\%\}) \parallel$$
$$(\text{stop}^\star[\text{role} = \text{``}super\text{''}](x)\sigma_3.Q_2 \ ,\{\text{bl} = 2\%\}) \parallel$$
$$(\text{stop}^\star[\top](x)\sigma_4.Q_3 \ ,\{\text{bl} = 2\%\})$$

Examples of interactions...

Broadcast synchronisation:

$$(\text{stop}^\star[\text{bl} < 5\%]\langle v\rangle\sigma_1.P \ ,\{\mathit{role} = \text{``}\mathit{master}\text{''}\}) \parallel$$
$$(\text{stop}^\star[\text{role} = \text{``}\mathit{master}\text{''}](x)\sigma_2 \ .Q_1 \ ,\{\text{bl} = 4\%\}) \parallel$$
$$(\text{stop}^\star[\text{role} = \text{``}\mathit{super}\text{''}](x)\sigma_3.Q_2 \ ,\{\text{bl} = 2\%\}) \parallel$$
$$(\text{stop}^\star[\top](x)\sigma_4.Q_3 \ ,\{\text{bl} = 2\%\})$$

$$\Downarrow$$

$$(P, \sigma_1(\{\mathit{role} = \text{``}\mathit{master}\text{''}\})) \parallel$$
$$(Q_1[v/x], \sigma_2(\{\text{bl} = 4\%\})) \parallel$$
$$(\text{stop}^\star[\text{role} = \text{``}\mathit{super}\text{''}](x)\sigma_3.Q_2, \{\text{bl} = 2\%\}) \parallel$$
$$(Q_3[v/x], \sigma_4(\{\text{bl} = 2\%\}))$$

Examples of interactions...

Broadcast synchronisation:

$$(\mathsf{stop}^{\star}[\mathsf{bl} < 5\%]\langle v\rangle\sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$

$$(\mathsf{stop}^{\star}[\mathsf{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathsf{bl} = 45\%\}) \parallel$$

$$(\mathsf{stop}^{\star}[\mathsf{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$

$$(\mathsf{stop}^{\star}[\top](x)\sigma_4.Q_3, \{\mathsf{bl} = 25\%\})$$

Examples of interactions...

Broadcast synchronisation:

$$(\mathsf{stop}^{\star}[\mathsf{bl} < 5\%]\langle v\rangle\sigma_1.P, \{role = \text{"}master\text{"}\}) \parallel$$
$$(\mathsf{stop}^{\star}[\mathsf{role} = \text{"}master\text{"}](x)\sigma_2.Q_1, \{\mathsf{bl} = 45\%\}) \parallel$$
$$(\mathsf{stop}^{\star}[\mathsf{role} = \text{"}super\text{"}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$
$$(\mathsf{stop}^{\star}[\top](x)\sigma_4.Q_3, \{\mathsf{bl} = 25\%\})$$

Examples of interactions. . .

Broadcast synchronisation:

$$(\mathrm{stop}^{\star}[\mathrm{bl} < 5\%]\langle v\rangle\sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$
$$(\mathrm{stop}^{\star}[\mathrm{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathrm{bl} = 45\%\}) \parallel$$
$$(\mathrm{stop}^{\star}[\mathrm{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathrm{bl} = 2\%\}) \parallel$$
$$(\mathrm{stop}^{\star}[\top](x)\sigma_4.Q_3, \{\mathrm{bl} = 25\%\})$$

$$\Downarrow$$

$$(P, \sigma_1(\{role = \text{“}master\text{”}\})) \parallel$$
$$(\mathrm{stop}^{\star}[\mathrm{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathrm{bl} = 45\%\}) \parallel$$
$$(\mathrm{stop}^{\star}[\mathrm{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathrm{bl} = 2\%\}) \parallel$$
$$(\mathrm{stop}^{\star}[\top](x)\sigma_4.Q_3, \{\mathrm{bl} = 25\%\})$$

Examples of interactions...

Unicast synchronisation:

$$(\mathsf{stop}[\mathsf{bl} < 5\%]\langle\bullet\rangle\sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathsf{bl} = 4\%\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$
$$(\mathsf{stop}[\top](x)\sigma_4.Q_3, \{\mathsf{bl} = 2\%\})$$

Examples of interactions...

Unicast synchronisation:

$$(\text{stop}[\text{bl} < 5\%]\langle \bullet \rangle \sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$
$$(\text{stop}[\text{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\text{bl} = 4\%\}) \parallel$$
$$(\text{stop}[\text{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\text{bl} = 2\%\}) \parallel$$
$$(\text{stop}[\top](x)\sigma_4.Q_3, \{\text{bl} = 2\%\})$$

Examples of interactions...

Unicast synchronisation:

$$(\mathsf{stop}[\mathsf{bl} < 5\%]\langle\bullet\rangle\sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathsf{bl} = 4\%\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$
$$(\mathsf{stop}[\top](x)\sigma_4.Q_3, \{\mathsf{bl} = 2\%\})$$

Examples of interactions...

Unicast synchronisation:

$$(\mathsf{stop}[\mathsf{bl} < 5\%]\langle\bullet\rangle\sigma_1.P, \{role = \text{“}master\text{”}\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathsf{bl} = 4\%\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$
$$(\mathsf{stop}[\top](x)\sigma_4.Q_3, \{\mathsf{bl} = 2\%\})$$

$$\Downarrow$$

$$(P, \sigma_1(\{role = \text{“}master\text{”}\})) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}master\text{”}](x)\sigma_2.Q_1, \{\mathsf{bl} = 4\%\}) \parallel$$
$$(\mathsf{stop}[\mathsf{role} = \text{“}super\text{”}](x)\sigma_3.Q_2, \{\mathsf{bl} = 2\%\}) \parallel$$
$$(Q_3, \sigma_4(\{\mathsf{bl} = 2\%\}))$$

Modelling the environment

Interactions between components can be affected by the environment:

- a wall can inhibit wireless interactions;
- two components are too distant to interact;
- . . .

Modelling the environment

Interactions between components can be affected by the environment:

- a wall can inhibit wireless interactions;
- two components are too distant to interact;
- ...

The environment...

- is used to model the intrinsic rules that govern the physical context;

Modelling the environment

Interactions between components can be affected by the environment:

- a wall can inhibit wireless interactions;
- two components are too distant to interact;
- ...

The environment...

- is used to model the intrinsic rules that govern the physical context;
- consists of a pair (γ, ρ):

Modelling the environment

Interactions between components can be affected by the environment:

- a wall can inhibit wireless interactions;
- two components are too distant to interact;
- ...

The environment...

- is used to model the intrinsic rules that govern the physical context;
- consists of a pair (γ, ρ):
  - a global store γ, that models the overall state of the system;

Modelling the environment

Interactions between components can be affected by the environment:

- a wall can inhibit wireless interactions;
- two components are too distant to interact;
- ...

The environment...

- is used to model the intrinsic rules that govern the physical context;
- consists of a pair (γ, ρ):
  - a global store γ, that models the overall state of the system;
  - an evolution rule ρ that regulates component interactions (receiving probabilities, action rates,...).

The evolution rule

It is assumed that all actions in CARMA take some time complete and that this duration is governed by an exponential distribution.

The evolution rule

It is assumed that all actions in CARMA take some time complete and that this duration is governed by an exponential distribution.

However the action descriptions do not include any information about the timing (unlike many other stochastic process algebras).

The evolution rule

It is assumed that all actions in CARMA take some time complete and that this duration is governed by an exponential distribution.

However the action descriptions do not include any information about the timing (unlike many other stochastic process algebras).

We also do not assume perfect communication, i.e. there may be a probability that an interaction will fail to complete even between components with appropriately match attributes.

The evolution rule

It is assumed that all actions in CARMA take some time complete and that this duration is governed by an exponential distribution.

However the action descriptions do not include any information about the timing (unlike many other stochastic process algebras).

We also do not assume perfect communication, i.e. there may be a probability that an interaction will fail to complete even between components with appropriately match attributes.

The environment manages these aspects of system behaviour, and others in the evolution rule.

The evolution rule ρ

ρ is a function, dependent on current time, the global store and the current state of the collective, returns a tuple of functions $\varepsilon = \langle \mu_p, \mu_w, \mu_r, \mu_u \rangle$ known as the evaluation context

- $\mu_p(\gamma_s, \gamma_r, \alpha)$: the probability that a component with store γ_r can receive a broadcast message α from a component with store γ_s;
- $\mu_w(\gamma_s, \gamma_r, \alpha)$: the weight to be used to compute the probability that a component with store γ_r can receive a unicast message α from a component with store γ_s;
- $\mu_r(\gamma_s, \alpha)$ computes the execution rate of action α executed at a component with store γ_s;
- $\mu_u(\gamma_s, \alpha)$ determines the updates on the environment (global store and collective) induced by the execution of action α at a component with store γ_s.

CARMA Operational Semantics

The operational semantics of CARMA specifications is defined in terms of three functions that compute the possible next states of a component, a collective and a system:

CARMA Operational Semantics

The operational semantics of CARMA specifications is defined in terms of three functions that compute the possible next states of a component, a collective and a system:

1. the function $\mathbb{C}$ that describes the behaviour of a single component;

CARMA Operational Semantics

The operational semantics of CARMA specifications is defined in terms of three functions that compute the possible next states of a component, a collective and a system:

1. the function $\mathbb{C}$ that describes the behaviour of a single component;
2. the function $\mathbb{N}_{\varepsilon}$ builds on $\mathbb{C}$ to describe the behaviour of collectives;

CARMA Operational Semantics

The operational semantics of CARMA specifications is defined in terms of three functions that compute the possible next states of a component, a collective and a system:

1. the function $\mathbb{C}$ that describes the behaviour of a single component;
2. the function $\mathbb{N}_{\varepsilon}$ builds on $\mathbb{C}$ to describe the behaviour of collectives;
3. the function $\mathbb{S}_{t}$ that shows how CARMA systems evolve.

Quantitative Analysis

The semantics of CARMA gives rise to a Continuous Time Markov Chain (CTMC).

This can be analysed by

- by numerical analysis of the CTMC for small systems;
- by stochastic simulation of the CTMC;
- by fluid approximation of the CTMC under certain restrictions (particularly on the environment).

A running example...

Bike Sharing System...

We want to use CARMA to model a bike sharing system where:

- bikes are made available in a number of stations that are placed in various areas of a city;
- Users that plan to use a bike for a short trip
  - can pick up a bike at a suitable origin station
  - return it to any other station close to their planned destination.
- we assume that the city is partitioned in homogeneous zones...
  - and that all the stations in the same zone can be equivalently used by any user in that zone.

CASL: Component Prototypes

The BSS scenario...

Two kinds of components, one for each of the two groups of agents involved in our BSS, can be considered:

- parking stations;
- users.

PS attributes:

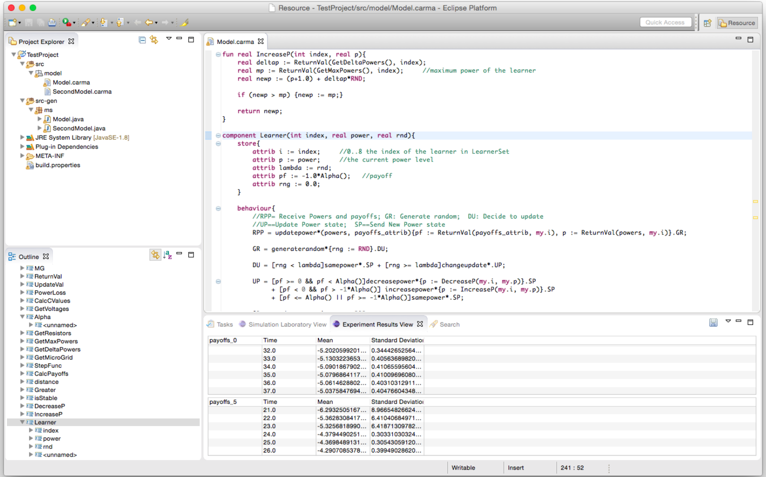
- **zone**: indicates where the station is located;
- **capacity**: the number of slots installed in the station;
- **available**: the number of available bikes.

User attributes:

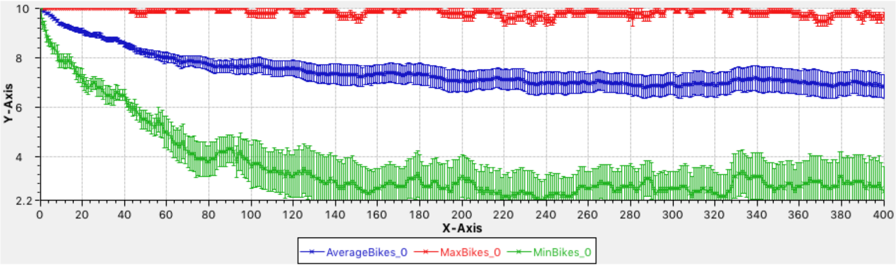
- **zone**: current user location;
- **dest**: user destination.

The CARMA Eclipse Plug-in

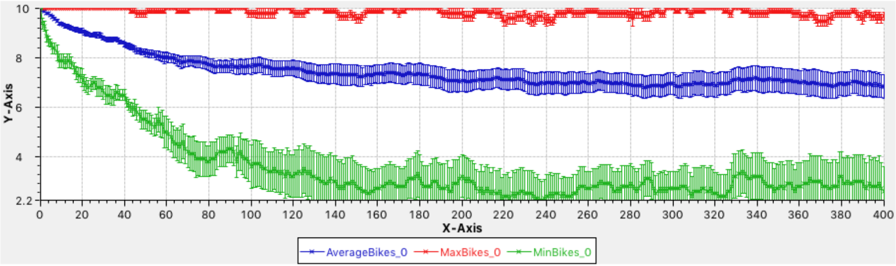
http://quanticol.sourceforge.net/



CASL: BSS Analysis



CASL: BSS Analysis



In this scenario the use of stations is not well balanced!

CaSL: an alternative model for BSS

To overcome this problem we can consider a model where stations located at the same zone do not compete but cooperate.

CASL: an alternative model for BSS

To overcome this problem we can consider a model where stations located at the same zone do not compete but cooperate.

We consider a variant of stations that, when located at the same zone, interact to avoid unbalanced use of resources.

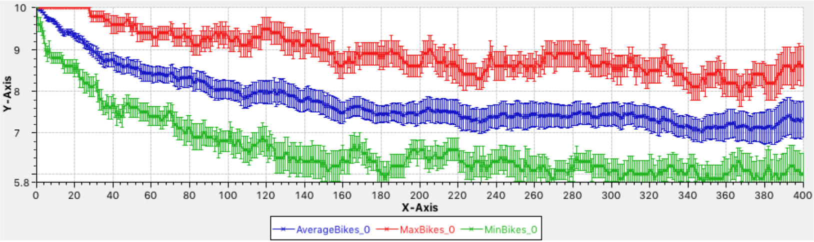
CaSL: an alternative model for BSS

To overcome this problem we can consider a model where stations located at the same zone do not compete but cooperate.

We consider a variant of stations that, when located at the same zone, interact to avoid unbalanced use of resources.

Each station can use broadcast to advertise other agents about the use of resources!

CASL: modified BSS model Analysis



Concluding remarks

- Collective Systems are an interesting and challenging class of systems to design and construct.

Concluding remarks

- Collective Systems are an interesting and challenging class of systems to design and construct.
- Their role within infrastructure, such as within smart cities, make it essential that quantitative aspects of behaviour are taken into consideration, as well as functional correctness.

Concluding remarks

- Collective Systems are an interesting and challenging class of systems to design and construct.
- Their role within infrastructure, such as within smart cities, make it essential that quantitative aspects of behaviour are taken into consideration, as well as functional correctness.
- The complexity of these systems poses challenges both for model construction and model analysis.

Concluding remarks

- Collective Systems are an interesting and challenging class of systems to design and construct.
- Their role within infrastructure, such as within smart cities, make it essential that quantitative aspects of behaviour are taken into consideration, as well as functional correctness.
- The complexity of these systems poses challenges both for model construction and model analysis.
- CARMA aims to address many of these challenges, supporting rich forms of interaction, using attributes to capture explicit locations and the environment to allow adaptivity.

thank you!