Stochastic Coordination in CAS:

Expressiveness & Predictability

Stefano Mariani   Andrea Omicini

{s.mariani, andrea.omicini}@unibo.it

DISI

Alma Mater Studiorum—Università di Bologna

"CAS: Qualitative and Quantitative Modelling and Analysis" Seminar
Dagstuhl, Germany - December 14-19th, 2014

Outline

Stochastic Coordination in CAS: Why

- CAS are usually composed of a *multitude of heterogeneous, autonomous agents*, which needs to interact in order to achieve a system goal—typically beyond their individual capabilities

  $\Longrightarrow$ coordination is a fundamental concern when *both analysing and modelling* CAS

- CAS often exhibit stochastic behaviours, stemming from *probabilistic* and *time-dependent* local (interaction) mechanisms

  $\Longrightarrow$ stochastic coordination models and languages provide such mechanisms

Expressiveness & Predictability

Understanding *expressiveness* of coordination languages and achieving *predictability* of stochastic coordination models is a fundamental step in the quest for designing well-engineered CAS.

Stochastic Coordination in CAS: Issues

Expressiveness from both the computer science and computer engineering points of view:

- science — formal expressiveness of *coordination languages* (to *observe* systems) [Bonsangue et al., 2003]
- engineering — *core coordination mechanisms* enabling expression of collective behaviours (to *build* systems) [Fernandez-Marquez et al., 2012b]

Predictability in terms of the *"local-to-global"* issue [Beal and Bachrach, 2006] of CAS

- design the *local* interaction mechanisms to achieve desired *global* behaviour

... (and the other way around: encode the global behaviour in terms of the local mechanisms available)

Stochastic Coordination in CAS: Goals

1. Devise out core coordination mechanisms enabling *self-organisation and adaptiveness* in CAS
2. Understand their expressiveness, both in terms of *formal comparison* of languages' (relative) expressive power and in terms of *enabled "behaviours"*
3. Ensure predictable (and controllable) relationships between such core (*local*) mechanisms and the overall coordinated CAS (*global*) behaviour exist

Outline

Outline

Motivation & Goals

- LINDA [Gelernter, 1985] features *don't know* non-determinism handled with a *don't care* approach:

  don't know which tuple among the matching ones is retrieved by a getter operation (`in`, `rd`) can be neither specified nor predicted

  don't care nonetheless, the coordinated system (e.g. the CAS) is designed so as to keep on working whichever is the matching tuple returned

LINDA & CAS

CAS may need to implement stochastic behaviours like *"most of the time do this, but sometimes do that"*.

A Core Coordination Mechanism

- We define uniform coordination primitives (`uin`, `urd`) – first mentioned in [Gardelli et al., 2007] – as the specialisation of LINDA getter primitives featuring *probabilistic non-determinism* [Mariani and Omicini, 2014]

The uLINDA Approach

Uniform primitives allow CAS designers to both specify and (*statistically*) predict the probability to retrieve one specific (kind of) tuple among a bag of matching tuples.

- Uniform primitives are (one of) the core coordination mechanisms enabling self-organisation in CAS: a *minimal construct* able (alone) to impact the *observable properties* of a coordinated system

On Predictability I

- In a sequence of getter operations, LINDA *don't know* non-determinism makes any prediction of the overall behaviour impossible
  - sequences of LINDA getter operations present no predictable distribution over time—abstracting away from specific implementations
- In a sequence of getter operations, uLINDA *probabilistic* (uniform) non-determinism makes predictions about the overall behaviour possible
  - sequences of uLINDA getter operations tend to exhibit a predictable uniform distribution over time

On Predictability II

A Dice in LINDA

- We define tuple space `dice`
- We represent a six-face dice as a collection of six tuples: `face(1)`, `..., face(6)`
- We roll a dice by `rd`-ing a `face/1` tuple from `dice`: `dice ? rd(face(X))`

! We do not obtain the (stochastic) behaviour of a dice: e.g., it may *reasonably* happen that rolling the dice 10^9 times always results in `X/1`

On Predictability III

A Dice in uLinda

- Again, we define tuple space `dice`
- Again, we represent a six-face dice as a collection of six tuples: `face(1), ..., face(6)`
- We roll a dice by `urd`-ing a `face/1` tuple from `dice`: `dice ? urd(face(X))`

! Now, we do obtain the (stochastic) behaviour of a dice: repeating several times a roll, the six faces will tend to converge towards a uniform distribution

On Expressiveness: “Load Balancing” I

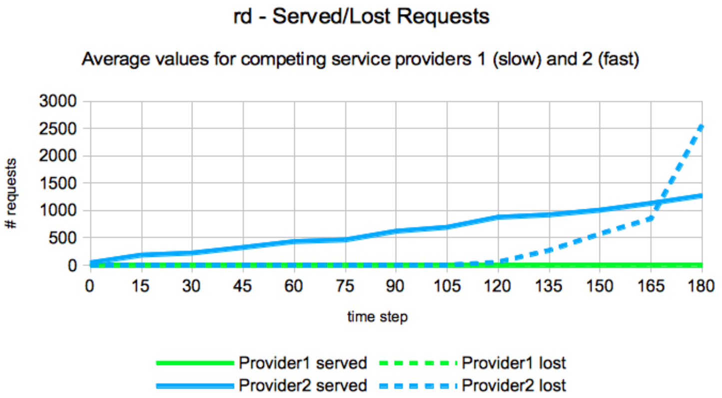

Figure : Clients using `rd`: `Provider1` is under-exploited—actually, not at all. Notice it could be `Provider2`, we don’t know “a priori”.

On Expressiveness: “Load Balancing” II

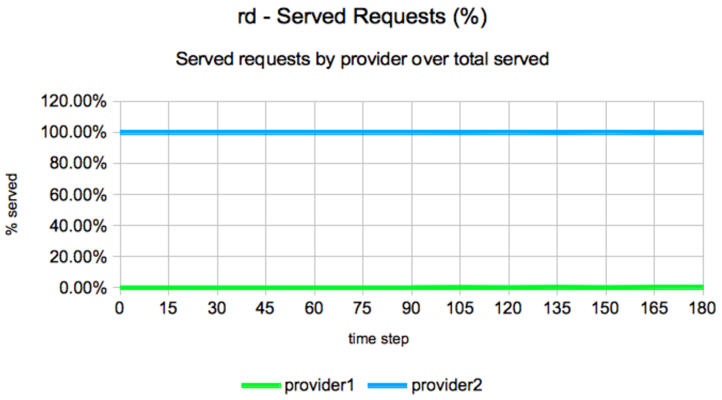

Figure : `Provider1` and `Provider2` exhibit no collective behaviour—no “load balancing”.

On Expressiveness: “Load Balancing” III

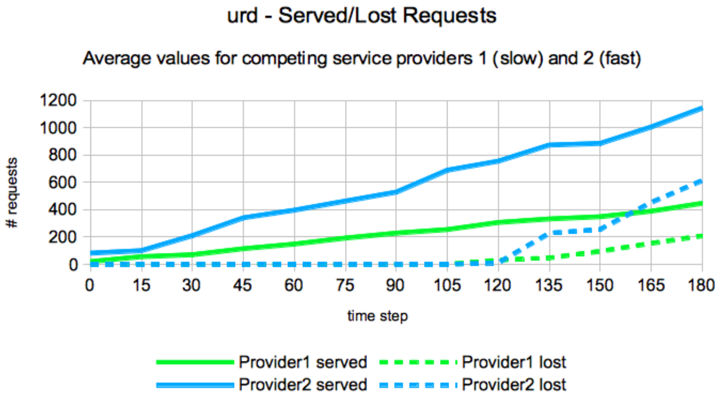

Figure : Clients using `urd`: `Provider2` is exploited as much as it can afford. `Provider1` takes what’s left.

On Expressiveness: “Load Balancing” IV

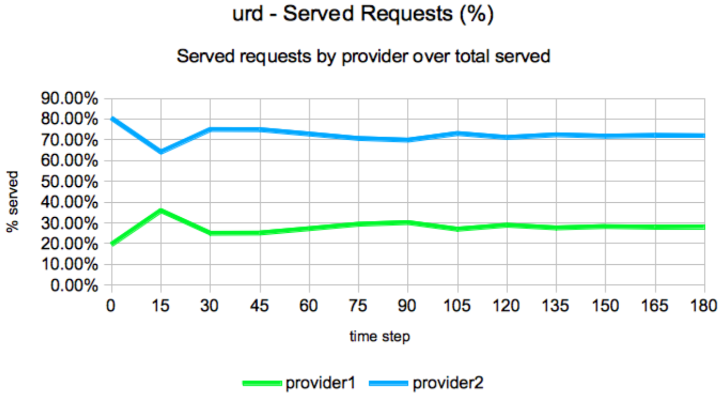

Figure : `Provider1` and `Provider2` exhibit some form of “load balancing”, achieved by self-organisation.

On Expressiveness: "Stigmergic Coordination" I

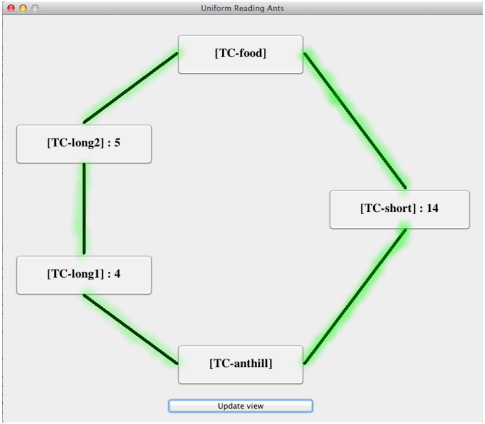

Figure : Digital ants search for food (top box) wandering randomly from their anthill (bottom box).

On Expressiveness: “Stigmergic Coordination” II

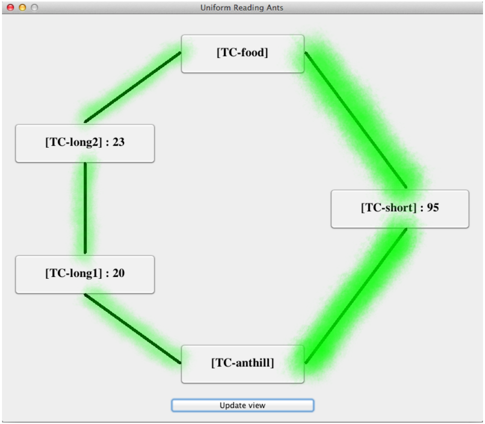

Figure : By `urd`-ing digital pheromones, ants find the optimal path toward the food source—as a self-organising, distributed process.

Outline

Motivation & Goals

- Understanding expressiveness of coordination languages is essential to deal with *interactions complexity* [Wegner, 1997]
- The notion of *Modular Embedding* [de Boer and Palamidessi, 1994] has been proposed for comparison of the relative expressiveness of *concurrent languages*

Expressiveness & CAS

CAS features stochastic behaviours, exploiting *probabilistic mechanisms*, which demands new techniques to model and *measure expressiveness*.

A New Measure of Expressiveness I

(Modular) Embedding

The informal definition of (modular) *embedding* assumes that a language could be translated in another [Shapiro, 1991]

- **easily** "without the need for a global reorganisation of the program"
- **equivalently** "without affecting the program's observable behaviour"

- We refine term "easily" to focus on coordination languages' expressiveness:
  - no *extra-computations* to mimic complex coordination operators
  - no *extra-coordinators* (neither coordinated processes nor coordination medium) to avoid suspensive semantics
  - no *unbound extra-interactions* to perform additional coordination

A New Measure of Expressiveness II

- We refine term "equivalently" to cover probabilistic coordination languages:
  - observable actions should be associated with their *execution probability*, driven by synchronisation opportunities offered by the coordination medium at run-time
  - ending states should be defined as those states for which all outgoing transitions have probability 0. Also, they should be refined with the probability of reaching that state from a given initial one

Probabilistic Modular Embedding

Using these refined definitions of "easily" and "equivalently"[a], we shift from Modular Embedding to Probabilistic Modular Embedding [Mariani and Omicini, 2013].

[a] It's a long story short, for the precise formalisation see the cited paper.

On Expressiveness I

- Suppose a ProbLinCa[1] process P and a LinCa process Q are acting on tuple space S:

$$P = \mathtt{in}_p(T).\emptyset + \mathtt{in}_p(T).\mathtt{rd}_p(T').\emptyset$$
$$Q = \mathtt{in}(T).\emptyset + \mathtt{in}(T).\mathtt{rd}(T').\emptyset$$
$$S = \langle \mathtt{t_l}[20], \mathtt{t_r}[10] \rangle$$

  where T is a LINDA template matching both tuples $\mathtt{t_l}$ and $\mathtt{t_r}$, whereas T' matches $\mathtt{t_r}$ solely.

[1] Probabilistic extension of the LINDA-based LinCa calculus, in which each tuple has a *weight* resembling selection probability: the higher the weight, the higher the matching chance [Bravetti et al., 2005].

On Expressiveness II

Modular Embedding

To ME, P and Q are *not distinguishable*, being their ending states the same:

$$\Psi[P] = (\texttt{success}, \langle \texttt{t}_\texttt{r}[10] \rangle) \text{ OR } (\texttt{deadlock}, \langle \texttt{t}_\texttt{l}[20] \rangle)$$
$$\Psi[Q] = (\texttt{success}, \langle \texttt{t}_\texttt{r}[10] \rangle) \text{ OR } (\texttt{deadlock}, \langle \texttt{t}_\texttt{l}[20] \rangle)$$

Quantity vs. Quality

Whereas P and Q are qualitatively equivalent, they are not so quantitatively, but ME cannot tell apart the probabilistic information conveyed by, e.g., a `ProbLinCa` primitive w.r.t. a `LinCa` one[a].

[a] Similar results can be obtained comparing pKLAIM to KLAIM and π_{pa}-*calculus* (probabilistic asynchronous pi-calculus) to π_a-*calculus* (asynchronous pi-calculus) [Mariani and Omicini, 2013].

On Expressiveness III

Probabilistic Modular Embedding

By repeating the embedding observation, now using PME, we get:

$$\Phi[P] = (0.\bar{6}, \texttt{success}) \text{ OR } (0.\bar{3}, \texttt{deadlock})$$
$$\Phi[Q] = (\bullet, \texttt{success}) \text{ OR } (\bullet, \texttt{deadlock})$$

where symbol $\bullet$ denotes "absence of information".

Quantity vs. Quality

PME succeeds in telling `ProbLinCa` apart from `LinCa`, recognising `ProbLinCa` as *more expressive* than `LinCa`, because it takes into account the probabilistic (quantitative) information conveyed by, e.g., a `ProbLinCa` primitive w.r.t. a `LinCa` one.

Outline

Motivation & Goals I

- Existing approaches to alleviate the so-called "local-to-global" issue [Beal and Bachrach, 2006] in CAS are mostly based on:
  - simulation [Gardelli et al., 2006]
  - parameter tuning [Gardelli et al., 2009]
  - (approximate) model checking [Casadei and Viroli, 2013]
  - "bio-inspired design patterns" [Fernandez-Marquez et al., 2012a]

Limits

- simulation may not be able to accurately reproduce real world contingencies
- parameter tuning may lead to sub-optimal settings
- model checking may be impractical for the complexity of the problem at hand
- design patterns give no guarantees about the quality of the solution

Motivation & Goals II

Goals

We propose an integrated approach [Mariani, 2014]:

1. rely on *design patterns* — design the local mechanisms by implementing self-organisation primitives as artificial chemical reactions
2. go beyond the *law of mass action* [Cardelli, 2008] — engineer custom kinetic rates for such reactions
3. "simulate-then-tune" [Gardelli et al., 2008] — adjust the dynamics of the (artificial) chemical system obtained (the CAS) to achieve the emergent, global behaviour desired

On Expressiveness I

- A survey of state-of-art literature[2] led to the following *core set of primitives*—the local mechanisms:

  decay destroys information as time passes

  feed increases information "relevance" (e.g. quantity) according to some kind of feedback mechanism

  activation/inhibition changes information "status" (e.g. attributes, values, etc.) depending on external stimuli

  aggregation fuses information together (e.g. filtering, merging, composing, transforming, etc.)

  diffusion moves information within a topology (e.g. migration, replication, etc.)

  repulsion/attraction drifts apart / approaches information

[2] [Nagpal, 2004], [De Wolf and Holvoet, 2007], [Fernandez-Marquez et al., 2012b], [Fernandez-Marquez et al., 2011], [Tchao et al., 2011], [Viroli et al., 2011]

On Expressiveness II

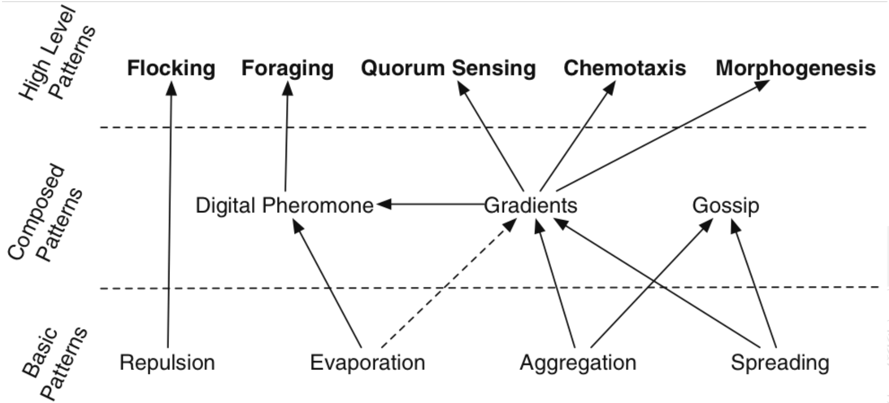

Figure : Bio-inspired design patterns, from *primitives* (local mechanisms) to high-level *patterns* (global behaviours). *Image taken from [Fernandez-Marquez et al., 2012a].*

On Predictability I

From Local to Global

By engineering the *local mechanisms* as artificial *chemical reactions*, and by engineering *custom kinetic rates*, we can relate changes in local mechanisms to the global behaviour of the CAS and observe/modify such relationships.

On Predictability II

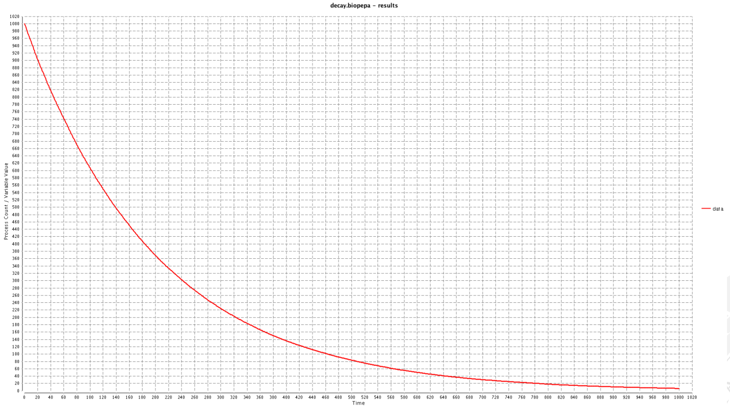

Figure : Decay chemical reaction, "fast-then-slow" trend, independent of the quantity of molecule to decay.

On Predictability III

Figure : Decay chemical reaction, “slow-then-fast” trend, now dependent on the quantity of molecule to decay.

On Predictability IV

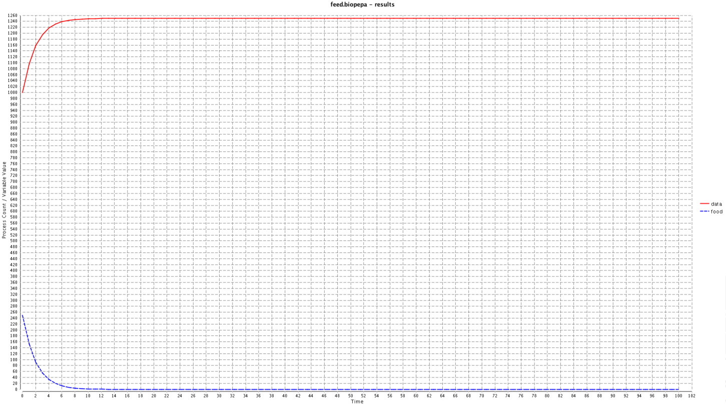

Figure : Feed chemical reaction, "fast-then-slow" trend, independent of the quantity of molecule to feed.

On Predictability V

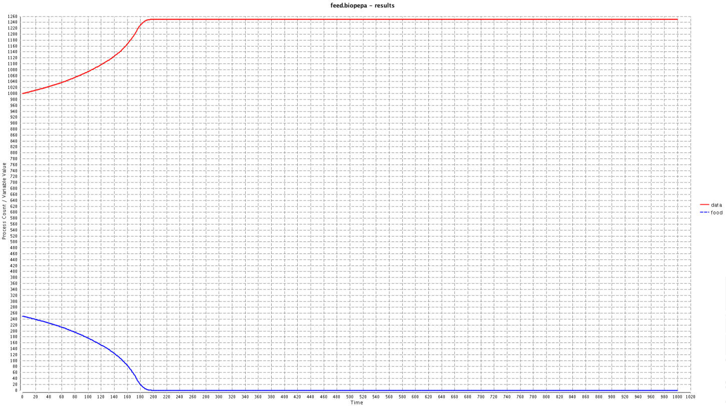

Figure : Feed chemical reaction, “slow-then-fast” trend, still independent of the quantity of molecule to feed.

On Predictability VI

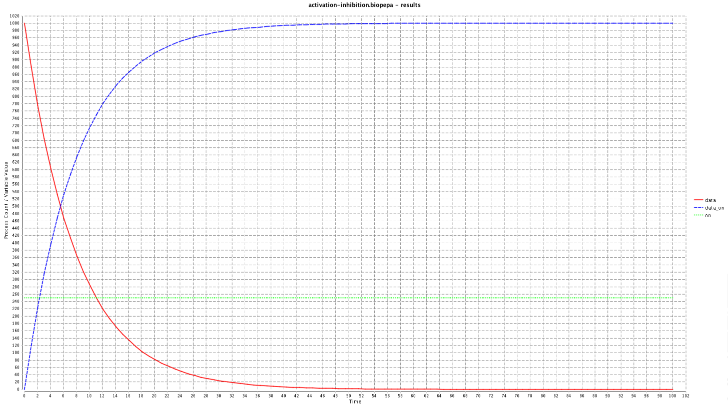

Figure : Activation chemical reaction, "fast-then-slow" trend, dependent on the quantity of activating molecule, independent of the quantity of molecule to activate.

On Predictability VII

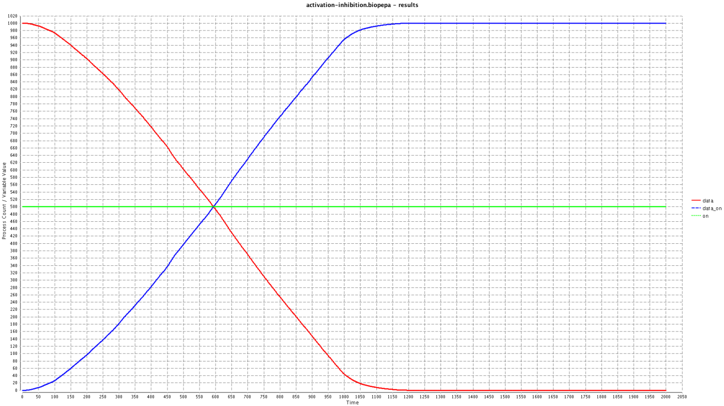

Figure : Activation chemical reaction, “nearly linear” trend, now independent of the quantity of activating molecule, now directly proportional to the quantity of molecule to activate.

On Predictability VIII

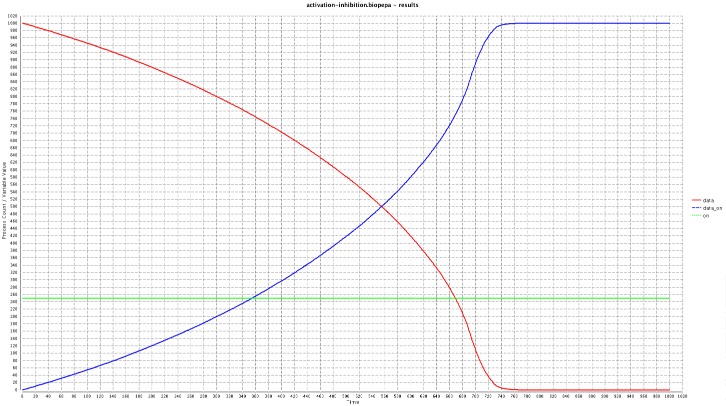

Figure : Activation chemical reaction, "slow-then-fast" trend, still independent of the quantity of activating molecule, still directly proportional to the quantity of molecule to activate.

On Predictability IX

Predictability

The factors chosen for custom kinetic rate expressions (the *local* mechanisms) have a well-defined, predictable effect on the *global* behaviour achieved by the CAS.

- This is made possible by adoption of the chemical reaction metaphor while implementing self-organisation primitives, and helps alleviating the "local-to-global" issue, ultimately leading to a better engineering of CAS behaviours

Outline

Future Directions I

Uniform Primitives Devising out *the whole spectrum* of self-organising behaviours they *directly enable* is essential to

- understand their modelling capabilities, from the perspective of a CAS designer
- understand their (practical) expressiveness, from the perspective of a CAS developer
- build a library of core coordination mechanisms enabling and supporting self-organisation in CAS

Future Directions II

PME Extending and further evaluating PME to *as many languages* as possible is essential to

- tell apart probabilistic languages into hierarchies of embeddings, devising out the core linguistic constructs from the “syntactic sugar”—in the spirit of [Vanglabbeek et al., 1995]
- define a novel notion of Turing Machine for CAS—in the spirit of [Wegner and Goldin, 2003]

Future Directions III

Biochemical Coordination Investigating effectiveness of artificial chemical reactions in *modelling bio-inspired design patterns* is essential to

- devise out expressiveness of the biochemical metaphor applied to coordination in CAS
- improve predictability of the *global* behaviours exhibited by the CAS

Open Questions

- ? How to know that a given set of *core* coordination mechanisms is complete? E.g. in the sense that it enables the *whole* spectrum of CAS self-organising behaviours?
- ? From the point of view of CAS engineering, what is the consequence of having different probability models placed in a *hierarchy* of embeddings? Does this imply a difference (even a hierarchy?) in the global behaviours they enable?
- ? Can we do what Wegner did with *Persistent Turing Machines* [Goldin, 2000], w.r.t. the interaction space of computations, in the field of CAS, building a theory of a Collective Adaptive Turing Machine?
- ? Is the biochemical metaphor scalable enough for large-scale CAS, e.g. world-wide socio-technical systems?

References I

Beal, J. and Bachrach, J. (2006).
Infrastructure for engineered emergence on sensor/actuator networks.
Intelligent Systems, IEEE, 21(2):10–19.

Bonsangue, M. M., Kok, J. N., and Zavattaro, G. (2003).
Comparing coordination models and architectures using embeddings.
Science of Computer Programming, 46:31 – 69.
Special Issue on Coordination Languages and Architectures.

Bravetti, M., Gorrieri, R., Lucchi, R., and Zavattaro, G. (2005).
Quantitative information in the tuple space coordination model.
Theoretical Computer Science, 346(1):28–57.

Cardelli, L. (2008).
On process rate semantics.
Theoretical computer science, 391(3):190–215.

References II

Casadei, M. and Viroli, M. (2013).
Toward approximate stochastic model checking of computational fields for pervasive computing systems.
In Pitt, J., editor, *Self-Adaptive and Self-Organizing Systems Workshops (SASOW)*, pages 199–204. IEEE CS.
2012 IEEE Sixth International Conference (SASOW 2012), Lyon, France, 10-14 September 2012. Proceedings.

de Boer, F. S. and Palamidessi, C. (1994).
Embedding as a tool for language comparison.
Information and Computation, 108(1):128–157.

De Wolf, T. and Holvoet, T. (2007).
Design patterns for decentralised coordination in self-organising emergent systems.
In *Engineering Self-Organising Systems*, pages 28–49. Springer.

Fernandez-Marquez, J., Marzo Serugendo, G., Montagna, S., Viroli, M., and Arcos, J. (2012a).
Description and composition of bio-inspired design patterns: a complete overview.
Natural Computing, pages 1–25.

References III

Fernandez-Marquez, J. L., Di Marzo Serugendo, G., and Arcos, J. L. (2011).
Infrastructureless spatial storage algorithms.
ACM Transactions on Autonomous and Adaptive Systems (TAAS), 6(2):15.

Fernandez-Marquez, J. L., Serugendo, G. D. M., and Montagna, S. (2012b).
Bio-core: Bio-inspired self-organising mechanisms core.
In *Bio-Inspired Models of Networks, Information, and Computing Systems*, volume 103 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 59–72. Springer Berlin Heidelberg.

Gardelli, L., Viroli, M., Casadei, M., and Omicini, A. (2007).
Designing self-organising MAS environments: The collective sort case.
In Weyns, D., Parunak, H. V. D., and Michel, F., editors, *Environments for MultiAgent Systems III*, volume 4389 of *LNAI*, pages 254–271. Springer.
3rd International Workshop (E4MAS 2006), Hakodate, Japan, 8 May 2006. Selected Revised and Invited Papers.

Gardelli, L., Viroli, M., Casadei, M., and Omicini, A. (2008).
Designing self-organising environments with agents and artefacts: A simulation-driven approach.
International Journal of Agent-Oriented Software Engineering, 2(2):171–195.
Special Issue on Multi-Agent Systems and Simulation.

References IV

Gardelli, L., Viroli, M., and Omicini, A. (2006).
On the role of simulations in engineering self-organising MAS: The case of an intrusion detection system in TuCSoN.
In Brueckner, S. A., Di Marzo Serugendo, G., Hales, D., and Zambonelli, F., editors, *Engineering Self-Organising Systems*, volume 3910 of *LNAI*, pages 153–168. Springer. 3rd International Workshop (ESOA 2005), Utrecht, The Netherlands, 26 July 2005. Revised Selected Papers.

Gardelli, L., Viroli, M., and Omicini, A. (2009).
Combining simulation and formal tools for developing self-organizing MAS.
In Uhrmacher, A. M. and Weyns, D., editors, *Multi-Agent Systems: Simulation and Applications*, Computational Analysis, Synthesis, and Design of Dynamic Systems, chapter 5, pages 133–165. CRC Press.

Gelernter, D. (1985).
Generative communication in Linda.
ACM Transactions on Programming Languages and Systems, 7(1):80–112.

Goldin, D. Q. (2000).
Persistent Turing Machines as a model of interactive computation.
In *1st International Symposium on Foundations of Information and Knowledge Systems (FoIKS '00)*, pages 116–135, London, UK, UK. Springer-Verlag.

References V

Mariani, S. (2014).
On the "local-to-global" issue in self-organisation: Chemical reactions with custom kinetic rates.
In *Proceedings of the 2014 Eigth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshop (SASOW 2014)*.

Mariani, S. and Omicini, A. (2013).
Probabilistic modular embedding for stochastic coordinated systems.
In Julien, C. and De Nicola, R., editors, *Coordination Models and Languages*, volume 7890 of *LNCS*, pages 151–165. Springer.
15th International Conference (COORDINATION 2013), Florence, Italy, 3–6 June 2013. Proceedings.

Mariani, S. and Omicini, A. (2014).
Coordination mechanisms for the modelling and simulation of stochastic systems: The case of uniform primitives.
SCS M&S Magazine.
Special Issue on "Agents and Multi-Agent Systems".

Nagpal, R. (2004).
A catalog of biologically-inspired primitives for engineering self-organization.
In *Engineering Self-Organising Systems*, pages 53–62. Springer.

References VI

Shapiro, E. (1991).
Separating concurrent languages with categories of language embeddings.
In *23rd Annual ACM Symposium on Theory of Computing (STOC'91)*, pages 198–208, New York, NY, USA. ACM.

Tchao, A.-E., Risoldi, M., and Di Marzo Serugendo, G. (2011).
Modeling self-* systems using chemically-inspired composable patterns.
In *Self-Adaptive and Self-Organizing Systems (SASO), 2011 Fifth IEEE International Conference on*, pages 109 –118.

Vanglabbeek, R. J., Smolka, S. A., and Steffen, B. (1995).
Reactive, generative, and stratified models of probabilistic processes.
Information and Computation, 121(1):59–80.

Viroli, M., Casadei, M., Montagna, S., and Zambonelli, F. (2011).
Spatial coordination of pervasive services through chemical-inspired tuple spaces.
ACM Transactions on Autonomous and Adaptive Systems, 6(2):14:1–14:24.

Wegner, P. (1997).
Why interaction is more powerful than algorithms.
Communications of the ACM, 40(5):80–91.

References VII

Wegner, P. and Goldin, D. (2003).
Computation beyond Turing machines.
Communications of the ACM, 46(4):100–102.

Stochastic Coordination in CAS:

Expressiveness & Predictability

Stefano Mariani   Andrea Omicini

{s.mariani, andrea.omicini}@unibo.it

DISI

Alma Mater Studiorum—Università di Bologna

"CAS: Qualitative and Quantitative Modelling and Analysis" Seminar

Dagstuhl, Germany - December 14-19th, 2014