Systemic Computation,
where natural computation and new technologies come together

Erwan Le Martelot
E&EE Department, University College London, London, U.K.
Context

Everything computes:

But:

<table>
<thead>
<tr>
<th>Nature computation properties</th>
<th>Computers computation properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stochastic</td>
<td>Deterministic</td>
</tr>
<tr>
<td>Asynchronous</td>
<td>Synchronous</td>
</tr>
<tr>
<td>Parallel</td>
<td>Serial</td>
</tr>
<tr>
<td>Homeostatic</td>
<td>Heterostatic</td>
</tr>
<tr>
<td>Continuous</td>
<td>Batch</td>
</tr>
<tr>
<td>Robust</td>
<td>Brittle</td>
</tr>
<tr>
<td>Fault tolerant</td>
<td>Fault intolerant</td>
</tr>
<tr>
<td>Autonomous</td>
<td>Human-reliant</td>
</tr>
<tr>
<td>Open-ended</td>
<td>Limited</td>
</tr>
<tr>
<td>Distributed</td>
<td>Centralised</td>
</tr>
<tr>
<td>Approximate</td>
<td>Precise</td>
</tr>
<tr>
<td>Embodied</td>
<td>Isolated</td>
</tr>
<tr>
<td>Circular Causality</td>
<td>Linear Causality</td>
</tr>
<tr>
<td>Complex</td>
<td>Simple</td>
</tr>
</tbody>
</table>

Fundamental differences between biological and traditional computation
Observations (1/3)

Issues in common technologies:

- Increasing performance, potential and complexity in machines and software
- Increasingly difficult to ensure reliability in systems
  - Software regularly crashes
  - Top of the line robots break down on the wrong kind of ground
  - Power distribution networks fail under unforeseen circumstances
  - ...

Particularly desirable natural properties:

- Adaptation
- Fault-tolerance
- Self-repair
- Self-Organisation
- Homeostasis (Varela, Maturana, and Uribe 1974)
Observations (2/3)

Computation is increasingly becoming more parallel, decentralised and distributed:

- Distributed computing (or multiprocessing)
- Computer clustering
- Grid computing
- Ubiquitous computing
- Speckled computing (Arvind and Wong 2004)
- ...
Observations (3/3)

- While hugely complex computational systems will be soon feasible, their organisation and management is still the subject of research (Kephart, Chess 2003):
  - Ubiquitous computing may enable computation anywhere,
  - Bio-inspired models may enable improved capabilities such as reliability and fault-tolerance,
  But no coherent architecture that combines both technologies.

- To address these issues and unify notions of biological computation and electronic computation, (Bentley 2007) introduced Systemic Computation.
Systemic Computation (1/2)

- Model of interacting systems with natural characteristics (Bentley 2007)

- New model of computation and corresponding computer architecture:
  - Systemics world-view
  - Incorporating natural characteristics

- Transformation is Computation

- Designed to support models and simulations of any kind of nature inspired systems, improving:
  - Fidelity,
  - Clarity
# Systemic Computation (2/2)

<table>
<thead>
<tr>
<th>1. Everything is a system</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="System Diagram" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. Systems interact within a context system</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image2" alt="System Interaction" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Systems can be transformed but never destroyed</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image3" alt="System Transformation" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. Systems may comprise or share other nested systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image4" alt="System Nesting" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5. Interactions are constrained by the scope of systems</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image5" alt="System Scope" /></td>
</tr>
</tbody>
</table>
Systemic Computer

System = Binary string divided into 3 parts
• 2 Schemata
• 1 Kernel

Non-Context systems
3 parts can hold anything (data, typing, etc.) in binary

Context systems
Kernel: Interaction result (optionally data)
Schemata: Allowable interacting systems

Example of interaction
Green: Context of Computation
Red: the Two Interacting Subjects
Systemic Computation Programming

SC program ≠ Conventional logic, procedural or object-oriented program

- Procedural program = sequence of instructions to process
- SC program = defines and declares a list of agents (the systems) in an initial state.
  - Execution: Let systems behave indefinitely and stochastically from their initial state
  - Outcome of the program resulting from an emergent process rather than a deterministic predefined algorithm.

```plaintext
label ANY ??
label MY_SYSTEM 0101
label MY_OTHER_SYSTEM 0111
label MY_K_DATA ??01
label MY_S_DATA 0111
label MY_CTX_DATA ??01

function NOP 00??
function My_Function 11??

system MySystem {
    MY_SYSTEM ,
    NOP | MY_K_DATA ,
    MY_S_DATA
}

system MyOtherSystem {
    MY_OTHER_SYSTEM ,
    NOP | MY_K_DATA ,
    MY_S_DATA
}

system MyContext {
    [ MY_SYSTEM , ANY, ANY ] ,
    My_Function | MY_CTX_DATA,
    [ MY_OTHER_SYSTEM , ANY, ANY ]
}

program {
    // Declarations
    Universe universe ;
    MySystem ms[1:2];
    MyOtherSystem mos[1:2];
    MyContext cs[1:2];

    // Scopes
}
```

Red : Universe
Yellow : MySystem
Blue : MyOtherSystem
Green : MyContext
Systemic Computation
An Alternative Approach

Conventional Program

- Infinite stochastic computation (systems randomly interact) and emergent program behaviour
- Clearly defined paradigm (7 rules, one entity – the system):
  - Method of modelling
  - Computer architecture
- Turing complete (Bentley 2007)
- Massively parallel computation
- Only local, asynchronous and independent computation
- Providing benefits for our understanding of "natural computing"
- Modelling of Genetic Algorithms, Neural networks (Le Martelot, Bentley, and Lotto 2007)
- Program with metabolism eating data and showing good abilities to detect anomalies in its diet (Le Martelot, Bentley, and Lotto 2008)

SC Program
Systemic Computation
An Alternative Approach

Program Robustness (Le Martelot, Bentley, and Lotto 2008): Fault tolerance & Self-repair

- Parallelism: a failed interaction does not prevent any further ones from happening
- Many independent systems: the failure of one of them cannot destroy the program
- No memory corruption at a global level, i.e. if a system contains fatal error, the program continues: The program is in a never ending loop
- Having multiple instances of similar systems also makes it easy to introduce a self-maintenance process with systems fixing each other
SC Actual Platform
Virtual Systemic Computer

(Le Martelot, Bentley, and Lotto 2007)

• The first high-level SC computer is a Virtual Machine (VM) running on a conventional PC.

• Dedicated programming language and associated compiler for the SC program.

• C++ code is used to write the context functions.

• VM runs program indefinitely from the initial state described in the program code
SC, Current Limits

SC designed to run bio-inspired systems natively
Inappropriateness of standard computers for increasingly complex systems

But
- VM remains limited by the underlying conventional architecture
- Two major limits: execution speed and the natural features the platform can provide

Speed issue: computer clustering (grid computing) ? peer-to-peer SC platform

Provide natural features natively:
- VM relies on the PC host’s brittle architecture and operating system compared to the robustness of natural computation.
- Grid computing and any form of clustering relying on conventional computers are also facing the same issue.
SC, New Technologies and Applications
Towards new implementations: Sensor Networks

SC implies a distributed, stochastic architecture, involving natural characteristics.

• Physical implementation of this architecture: wireless devices produced for sensor networks (Bentley 2007).
→ Provide an architecture for a useful, fault-tolerant and autonomous computer:
  - exploit all the features of sensor networks (e.g. data-centric paradigm),
  - robust wireless networking.

• Speckled computing (Arvind, Wong 2004): “spray on” computer coatings.
• Predictions of the logical extension to the Internet: wireless dynamic networks of computers in almost every device around us.

→ Technology of computation is increasingly becoming more parallel, decentralised and distributed, yet, such trends towards distributed and ubiquitous computing come at the cost of control and reliability.
→ Here SC offers a novel way to control computation defining rules working locally and providing emergent global properties such as fault-tolerance and self-repair.
SC, New Technologies and Applications
Towards new implementations: FPGAs (1/2)

• In SC, any system can be a context
• Interactions defined by contexts can change over computation → on-line reprogrammable devices.

Reconfigurable computing paradigm combines:
  • software flexibility
  • hardware high performance
into devices commonly called reconfigurable hardware, one of the most powerful being known as the Field-Programmable Gate Array (FPGA).

FPGA: a programmable logic component that can be reconfigured into different circuits (the configuration stored on its own internal memory)
  • computer processor
  • any other logic-based electronic circuit, with the ability to reconfigure its own circuits as it runs.
SC, New Technologies and Applications
Towards new implementations: FPGAs (2/2)

If reconfigured as a massively parallel architecture where
  • the routing between the logic blocks is dynamically changed
  • the computation reprogrammed on-line
→ Platform suitable for SC

However, limit in the amount of systems defined by the amount of inner logic blocks.

Ideal FPGA: hundreds of millions of logic blocks, with for each the possibility to connect anywhere in 3 dimensions like the brain.
→ Unfortunately, such technology is not available to date.

These limitations also apply for wireless networks limited by
  - Bandwidth
  - Interference
    - FLASH memory size
→ Unfortunately, not available to date either.
SC, New Technologies and Applications
Towards new implementations: Other devices

Other specific hardware have been designed to integrate natural features:

• POEtic project (Tempesti et al. 2002), aiming at creating a platform organised with a similar hierarchy as found in biological systems

• Perplexus (Upegui et al. 2007)

• Embryonics project (Tempesti et al. 2007) is also another project that investigated such hardware for reliability, involving self-replicating hardware.
SC, New Technologies and Applications

Potential Applications

- Based on a wireless network between objects or sensors:
  - Phone communications,
  - Radio transmission,
  - Digital audio players,
  - Internet of Things,
  - Smartdust,
  - Ambient Intelligence.

- Bio-inspired computing with SC hardware: fast-bio-inspired computation (e.g. NP-problems) not facing the limitations of conventional computers.

- Bio-inspired computation with SC can also offer good efficiency for data flow analysis and anomaly detection (e.g. Le Martelot, Bentley, and Lotto 2008b): Secure networks and reliable data transmission or control protocols (e.g., cognitive radio).
SC, New Technologies and Applications

Potential Applications

• Technologies closely linked to biological computation:
  - Nanotechnologies,
  - Self-assembling devices, evolving towards artificial molecular-level devices and machines (Balzani et al. 2002)

• Computation using biology or natural systems:
  - DNA-computing (Adleman 1994),
  - Bacteria-computing (Gardner 2000),
  - Reaction-diffusion computing (Adamatzky et al. 2002).

• World of robotics: Swarm robotics, using emerging collective behaviour like in an insect or an ant colony is another field where the SC paradigm would fit.

• SC: clearly defined framework for complex systems: improve the analysis and the understanding of complex systems (Bentley 2007)
Biological computation has many significant characteristics that help give it desirable properties.

Systemic Computation Paradigm:
- Novel and unconventional computation
- Unification of biological computation and electronic computation behind a coherent paradigm and computer architecture involving
  - Nature-inspired rules
  - Important properties such as fault-tolerance, self-repair
- Suitable for any kind of bio-inspired model
- Novel computer architecture

Implications for new hardware and technologies
- Compatibility with current electronics hardware
- SC Contribution to reliability and control in such systems
- Applicative potential from biological computation or bio-inspired computation to nanotechnologies, and daily applications like the internet or network security

• Adleman, L. M., Molecular Computation Of Solutions To Combinatorial Problems, Science 266 1021–1024,1994


References (2/4)


References (3/4)


