Unconventional Computation: Quo Vadis?

March 21-23, 2007
La Fonda on the Plaza Hotel
Santa Fe, New Mexico, USA

Conference Proceedings

Wednesday, March 21, 2007

Conference Talks
8:25 AM – 5:30 PM

Poster Session with Reception
5:45 PM – 8:00 PM

Thursday, March 22, 2007

Conference Talks
8:30 AM – 12:10 PM

Computation in the Brain Session
2:00 PM – 6:00 PM

Banquet
6:30 – 9:00

Friday, March 23, 2007

Conference Talks
8:30 AM – 5:15 PM

Organizers:
Christof Teuscher, Francis J. Alexander, Ilya Nemenman, & Gregory Johnson (LANL); Chris Wood (Santa Fe Institute)

The journal Physica D will publish a special issue after the workshop with selected contributions from both invited speakers and the contributors. Contributions are by invitation only and will be peer reviewed just like any regular submission to the journal.

Related Upcoming Conferences:
- Conference on Unconventional Computing, Bristol, UK, Jul 12-14, 2007
  http://uncomp.uwe.ac.uk/uc2007/
- 6th Int. Conference on Unconventional Computation, Aug 13-17, 2007
  http://www.cs.queensu.ca/uc07/

Image Credits: Illustrations kindly contributed by De Lacy Costello (c) 2005; Bachrach & Beal (c) 2006; Coore (c) 1999.

Center for Nonlinear Studies

Los Alamos NATIONAL LABORATORY EST. 1943

SANTA FE INSTITUTE Celebrating 20 years of Complexity Science

Wednesday, March 21

08:25-08:40	**Welcome** by Stephen R. Lee, Computer, Computational & Statistical Sciences (CCS) Division Leader
08:40-09:20	**Eric Drexler** *Unconventional Fabrication From Bits to Atoms and Back Again*
09:20-10:00	**Erik Winfree** *In vitro synthetic biology: from self-assembly to biochemical circuits*
10:00-10:30	Coffee
10:30-11:10	**Klaus-Peter Zauner** *Biological Computing Substrates*
11:10-11:30	**Ferran D. Revilla and Hywel Morgan and Klaus-Peter Zauner** *Physarum Polycephalum on a Chip for Bio-hybrid Information Processors*
11:30-12:10	**Andy Adamatzky** *Computing by propagating patterns:* *semantics of reaction-diffusion and Physarum machines*
12:10-02:00	Lunch *(on your own)*
02:00-02:40	**Michael L. Simpson** *Cell-Like Computation in Complex Synthetic Nanoscale Systems*
02:40-03:20	**David H. Wolpert** *On the computational capabilities of physical systems*
03:20-03:50	Coffee
03:50-04:30	**Norman Margolus** *Both Continuous and Discrete*
04:30-05:10	**Leon Chua** *CNN: A Brain-like Computer on a Chip*
05:10-05:30	**Maria-Magdolna Ercsey-Ravasz** *Cellular neural network computing in physics*
05:45-08:00	Poster session and reception

Unconventional Fabrication From Bits to Atoms and Back Again

K. Eric Drexler
Nanorex

Fabrication based on mechanically directed molecular manipulation can be regarded as an unconventional form of computation in which the output pattern is valued more for its physical embodiment than for its information content. Indeed, there are fundamental physical parallels between this mode of molecular fabrication and classical digital logic. Regarding applications, advanced molecular fabrication methods will enable the production of an unprecedented range of atomically precise structures, providing new means for implementing unconventional computational systems. Developments in molecular nanotechnologies, including DNA engineering, are steps toward implementing molecular fabrication systems of this kind.

Among the similarities between classical digital logic and directed molecular manipulation are discreteness of states, noise margins separating those states, and the consequent potential for exponentially low error rates. Shared physical issues include thermal fluctuations and the role of bit-erasure and dissipative driving forces in determining energy efficiency. Both computing and fabrication systems have potential implementations based on functional components of nanometer scale and high operating frequency. Each provides the basis for a transition from specialized analog systems to programmable systems with new and broad capabilities.

Among the differences, however, are the diversity of non-substitutable operations in fabrication (where there is nothing analogous to NAND or NOR) and the absence of a crisp concept of universality. A difference of great practical importance is the asymmetry of their mutual causal relationships: Fabrication systems can make both fabrication and computing systems, while computing systems can make neither. Substituting "design" for "fabrication", these roles are reversed.

Many unconventional computing systems will require unconventional structures if they are to be implemented. In many instances, intricate structures, small devices, and great precision will be either desirable or necessary. Molecular fabrication systems can enable the realization of a broad class of structures of this sort, in a wide range of materials, with components manufactured to complex, atomic specifications. This prospect will, I hope, suggest possibilities that encourage investigation of novel concepts and physical embodiments for unconventional computing. (For example, I'd like to see theoretical exploration of computing devices that exploit atomically precise structures and competing electronic domains in strongly correlated electronic materials.)

Currently, progress in molecular nanotechnologies is dominated by self-assembly. Structural DNA nanotechnologies are now able to routinely fabricate precise, million-atom, 100-nm-scale structures with hundreds of independently addressable binding sites for other components. Exploiting this technology to provide frameworks for modular composite nanosystems offers a path toward technologies for both molecular fabrication and unconventional computing.

In vitro synthetic biology: from self-assembly to biochemical circuits

Erik Winfree
Caltech

Biological organisms are a beautiful example of programming. The program and data are stored in biological molecules, and the algorithms are carried out by molecular and biochemical processes. If we understood how to program biomolecular systems, what could we create? Molecular machines, systems, an artificial cell, from scratch? In this talk I will discuss on our lab's attempts to get our heads around these questions by focussing on information and information processing in molecular systems. Three examples will be how to program the self-assembly of DNA tiles to create algorithmically patterned crystals, how to program in vitro transcription networks to obtain dynamical behavior, and how to program nucleic acid interactions to create digital logic circuits.

Biological Computing Substrates

Klaus-Peter Zauner

University of Southhampton, UK

A crucial difference sets apart present computing technology from information processing mechanisms utilised by organisms: The former is based on formalisms which are defined in disregard of the physical substrate used to implement them, while the latter directly exploit the physico-chemico properties of materials. There are many advantages to isolating the operation from implementation as is the case in current computers---but theses come at the cost of low efficency. In applications where size and energy-consumption is tightly restricted, or where real-time response to ambiguous data is required organisms cope well, but existing technology is unsatisfactory.

Taking heed of the clues from biology the question arises how the realm of computer science can be extended from formal to physical information paradigms. The aim is to arrive at a technology in which the course of computation is driven by the physics of the implementation substrate rather than arbitrarily enforced. The traditional tools and approaches of computer science are ill suited to this task. In particularly it will be necessary to orchestrate autonomously acting components to collectively yield desired behaviour without the possibility of prescribing individual actions.

Bio-electronic hybrid systems can be serve as a starting point to explore approaches to computing with autonomous components. We take a two-pronged approach in which we recruit both molecules and complete cells as biological computing substrate. Molecules offer reproducible nonlinearity, self-assembly, and high integration density of complex input-output mappings. Cells, on the other hand, provide cheap and fast nano-engineering through self-reproduction, build in quality-assurance through testing at the point of assembly, self-reconfiguration, and self-repair. Molecules, however, require infrastructure and cells are typically too complex for efficient computation. Our expectation therefore is that in the long term practical biological computing substrates will be situated at the subpramolecular and subcellular level, i.e., at the interface between inanimate and animate matter.

Physarum Polycephalum on a Chip for Bio-hybrid Information Processors

Ferran D. Revilla, Hywel Morgan and Klaus-Peter Zauner

University of Southhampton, UK

The true slime mold Physarum polycephalum plasmodium, a giant amoeboid cell, has raised interest for its information processing capabilities [1]. Despite a the lack of a centralised control organ, P. polycephalum is able to integrate information from different sources (temperature, light, chemical environment) and different areas of its surface and is capable of generating a global and seemingly smart response to adapt to its environment [2]. P. polycephalum can be seen as a highly distributed natural computing system in which signals are conveyed among parts of the plasmodial cell through the propagation of contraction waves along a mesh of intracellular veins.

In ongoing work towards integrating this organism into bio-hybrid devices we have created a novel micro-device. A plasmodium is enclosed by a thin gas-permeable layer of poly-dimethyl-siloxane (PDMS) on the patterned copper side of the PCB and by a polycarbonate filtration membrane on the reverse. A micro-fluidic system, manufactured from PDMS, is clamped to the top of filtration membrane to provide both a suitable environment to sustain the slime mold for over a week, and to permit localised stimulation with chemical signals. Furthermore, this micro-chip with a size of 20 by 25 mm also enables the monitoring of the of the intracellular oscillations implicated in P. polycephalum's information integration [3]. To this end electrodes etched into the PCB and sealed with photo-resist are used for impedance measurements. By performing simultaneously optical density measurements and impedance measurements we have verified that the impedance measurement technique provides an alte! rnative possibility to interface with the plasmodium.

The micro-chip is useful in the study and characterisation of natural computation and information integration processes in P. polycephalum. It also provides a potential route for integrating plasmodia into devices such as living sensors or robot controllers [4].

[1] S. Tsuda, M. Aono, Y.-P. Gunji, Robust and emergent physarum logical-computing, BioSystems 73 (2004) 45-55. [2] Y. Miyake, S. Tabata, H. Murakami, M. Yano, and H. Shimizu, Environment-Dependent Self-Organization of Positional Information Field in Chemotaxis of Physarum Plasmodium Journal of Theoretical Biology (1996) 178, 341-353 [3] Y. Miyake, H. Tada, M. Yano and H. Shimizu, Relationship between Intracellular Period Modulation and External Environment Change in Physarun Plasmodium, Cell structure and function, (1994) 19, 363-370. [4] S. Tsuda and K.-P. Zauner, Y.-P. Gunji, Robot Control with Biological Cells, Biosystems, 87 (2007) 215-223.

Computing by propagating patterns: semantics of reaction-diffusion and Physarum machines

Andrew Adamatzky

University of the West of England, UK

A paradigm of reaction-diffusion computing states that being suitably encoded in concentration profiles of a chemical medium any problem can be, in principle, solved by propagating and interacting chemical and phase waves. There is a huge gap however between constructing a couple of logical gates in a medium, and embedding a realistic logical circuit, or solving a large-scale optimization problem of the real world. Therefore in our talk we aim to fill the gap by adopting a goal-oriented approach in designing non-linear medium computers. To start with we show how famous problems of computational geometry (plane tessellation and skeletonisation) and robot control (taxis and obstacle avoidance) can be solved in linear time in reaction-diffusion chemical systems. We then establish a deficiency of the reaction-diffusion computing devices by demonstrating that it is impossible to invert Voronoi diagram, approximate minimum spanning tree and shortest path in neither excitable nor precipitating chemical systems without bringing up external computing structures. To cure the computational inadequacy we should allow just few points of a wave-front to guide propagation of the computing patterns. Such a constraint is realized in a slime mold, which is enclosed in a membrane analogue of reaction-diffusion media. We provide experimental evidences that Physarum slime mold interactively computes a succession of proximity graphs: nearest neighbour, spanning tree, relative neighbourhood and Gabriel graphs. Final part of the talk addresses programmability of non-linear medium computers, semantics of reaction-diffusion computing, based on Hagiya automata rules, and interpretation of Physarum computing in terms of Kolmogorov-Uspenskii machines.

Cell-Like Computation in Complex Synthetic Nanoscale Systems

M. L. Simpson

Oak Ridge National Laboratory

In a landmark paper published in Science in 1972, Anderson observed "that at each level of complexity entirely new properties appear, and the understanding of the new behaviors require research which I think is as fundamental in it nature as any other". This observation highlights a *grand Challenge* facing the nanoscience community: moving beyond just the synthesis of individual or homogeneous arrays to complex arrangements of nanoscale materials that lead to new classes of functionality. Responding to this grand challenge presents insurmountable obstacles for the classical approach of reductionism that works to understand phenomena in segments rather than as a system, when confronting the twin difficulties of scale and complexity. Instead, what is required is the rethinking of the usual paradigm of research flowing from synergies between theory, synthesis, and characterization, as there is no general theory to guide the synthesis and organization of nanoscale materials into highly interactive collectives. I will discuss a research paradigm that includes a fourth component – the top-down observation of the organization and associated function of natural complex systems, and the transfer of these principles into the bottom-up synthesis of complex synthetic nanoscale systems. The best examples of high levels of functionality emerging from ensembles of nanoscale elements are found in biological cells that perform extremely complex functions that include sensing, communication, navigation, cooperation, and even fabrication and organization of synthetic nanoscale materials, which undoubtedly include computation on many different levels of the system. There are striking differences when synthetic and natural nanoscale systems are compared, and these difference are found both in scale (density of functional elements) and complexity (density of interactions between the elements), and it follows that nanoscience efforts should focus both on the synthesis of nanomaterials (scale) and their organization into ensembles (complexity). The philosophical underpinning of this approach is well-captured in a phrase attributed to Feynman: ***"What I cannot create I do not understand."*** This talk will focus on this connection between design and understanding as it relates to nanoscale systems that lead toward cell-like complexity in synthetic systems.

On the computational capabilities of physical systems

David H. Wolpert

NASA Ames

In this talk strong limits on computation in the physical universe are presented. These limits are based on a novel definition of computation, ``inference devices". Inference devices model actual physical computers, in contrast to Chomsky hiearchy constructions like Turing machines.

1) There cannot be an inference device to which one can pose all computational tasks concerning the universe.

2) No inference device can correctly carry out any computational task in the subset of such tasks that can be posed to it.

In particular, no inference device can take the state of an arbitrary system external to that device as input, and correctly predict that system's future state before that future state actually occurs. The result also mean that there cannot exist an infallible, general-purpose observation apparatus. It also means that there cannot be an infallible, general-purpose control system.

(1) and (2) do not rely on external systems that are infinite, and/or non-classical, and/or obey chaotic dynamics. They also hold for an infinitely fast, infinitely dense inference device, with computational powers greater than that of a Turing Machine.

3) There are impossibility results for inference devices that are analgous to Chomsky hierarchy results concerning universal Turing Machines and the Halting theorem, and results concerning the (im)possibility of certain kinds of error-correcting codes.

4) There is an analogue of algorithmic information complexity for inference devices, ``prediction complexity".

A task-independent bound exists on how much the prediction complexity of a computational task can differ for two different universal inference devices used to solve that task. This bound is similar to the ``encoding" bound governing how much the algorithm information complexity of a Turing machine calculation can differ for two universal Turing machines.

5) Either the Hamiltonian of our universe prohibits a certain type of computation, or prediction complexity is unique (unlike algorithmic information complexity).

Both Continuous and Discrete

Norman Margolus

MIT

In computational Physics, continuous motion is usually regarded as a limit approached as resolution in space and time is increased. An unconventional alternative is to employ computational models which exactly reproduce a finite-space and finite-time sampling of the behavior of a continuous dynamics. Fredkin's Billiard Ball Model is a well known example of a system of this sort: with constrained initial conditions and when viewed at integer times, the continuous classical mechanical time evolution of colliding hard spheres exactly reproduces a discrete digital dynamics.

This talk is about these sorts of unconventional computational models, which "obey" a continuous dynamics. Such models are interesting from the viewpoint of theoretical physics. Like quantum systems, they have both continuous and discrete aspects. They make contact between the continuous language of physics and the discrete language of computation, allowing continuum-physics ideas such as momentum and energy to be directly carried over into the discrete world. They also provide examples of finite-state systems that obey classical mechanics, and so provide a classical substratum for counting states both in statistical mechanics and in classical dynamics.

CNN : A Brain-like Computer on A Chip

Leon Chua

University of California, Berkeley

CNN is an acronym for Cellular Nonlinear Networks. It is endowed with a non-von Neumann architecture reminiscent of von Neumann's Cellular Automata but computes via continuous flows along vector fields. It is also endowed with the key feature of a von Neumann architecture in that it is fully programmable via an user-friendly C-like language. It is a massively parallel processing machine where each cell is a caricature of a neuron , and where computation is both brain-like and emergent , made possible via the principle of local activity. All these unconventional features had been encapsulated in a tiny 176 x 144 silicon chip , thereby providing an enabling technology for many mission-critical tasks currently solvable only by supercomputers.

Cellular neural network computing in physics

Maria-Magdolna Ercsey-Ravasz

Babes-Bolyai University, Cluj-Napoca

Unconventional computing has been a very important research topic since it became clear that the power of conventional digital computers can not grow much further with the same speed as it did in the last decades. One of the many new computational paradigms exploited is cellular neural network computing, many times called also as cellular wave computing. This is based on the theory of cellular neural/nonlinear networks (CNN) [1] and is experimentally realized by different physical principles in the architecture of the CNN Universal Machine (CNN-UM) [2]. This new computational paradigm is well suited for some special, very complex problems, and the CNN-UM chips can be used expletively with conventional computers.

Possibilities for performing fast image processing, developing different sensors, solving differential equations, and cellular automata models were already studied. Here we show that using the fast realistic random number generator developed on the CNN-UM presented in [3] a new - very broad - class of problems of physics can be handled on this computer. These problems are mainly lattice models, which may require random initial conditions or stochastic (Monte Carlo type) dynamics. Two basic problems are studied: the two-dimensional site-percolation problem and the two-dimensional Ising model. Both represent a whole class of problems and the algorithms developed could be easily changed for many kindred problems: like bond percolation, directed percolation, diluted Ising model etc. The algorithms were tested on an experimental version of the CNN-UM (ACE16K chip), which has 128 * 128 cells. The results are in good agreement with results obtained on digital computers. Time measurements suggest that developing trend of the CNN-UM chips - increasing the lattice size, and appearance of three dimensional chips - will assure an important advantage for the CNN-UM in future.

[1] L. O. Chua, L. Yang, "Cellular neural networks: Theory", IEEE Transactions on Circuits and Systems, vol. 35, p. 1257, (1988) [2] T. Roska, L. O. Chua, "The CNN Universal Machine: an analogic array computer", IEEE Transactions on Circuits and Systems, II, vol. 40, p. 163, (1993) [3] M. Ercsey-Ravasz, T. Roska, Z. Neda, "Perspectives for Monte Carlo simulations on the CNN-UM", Int. J. of Modern Physics C, Vol. 17, No. 6, p. 909, (2006)

Thursday, March 22

08:30-08:35 Announcements

08:35-09:15 **Steen Rasmussen**
Self-replicating materials, information processing, and the Omega-Machine

09:15-09:55 **Darko Stefanovic**
Biomolecular Automata Using Deoxyribozymes: Accomplishments and Open Problems

09:55-10:25 Coffee

10:25-11:05 **Chris Dwyer**
DNA Self-assembly and Computer System Fabrication

11:05-11:50 **Howard Barnum**
Quantum Computation: How Goes It?

11:50-12:10 **Karoline Wiesner, John Mahoney, and James P. Crutchfield**
Intrinsic Computation in Quantum Processes: An Information-Theoretic Analysis Using Quantum Finite-State Machines

12:10-02:00 Lunch *(on your own)*

02:00-06:00 ***"Computation in the Brain,"*** **organized by Chris Wood, Santa Fe Institute**
See insert for more information

06:30-09:00 Conference Banquet in La Terraza Room

Self-replicating materials, information processing, and the Ω-Machine

Steen Rasmussen

LANL

Self-replicating materials could presumably provide important properties for novel computing substrates, in particular if the self-replicating materials could communicate with conventional digital computing substrates.

We discuss recent advances in assembling self-replicating materials from scratch as we have demonstrated experimentally (DeClue et al., 2007) that an informational molecule can catalyze the metabolic production of containers, connecting the main components of minimal life in a protocell (Rasmussen et al., 2004). Theoretical results predict that self-replicating informational molecules catalytically coupled to the existing growing container system (integration not yet experimentally implemented) should ensure balanced reproduction of all the protocellular components when integrated (Rocheleau et al., 2007 & Munteneau et al., 2007).

The objective of the so-called Ω-machine (PACE) is to provide a computer programmable parallel microfluidics support system for the evolution of protocells and other complex chemical systems connecting digital computing and chemical information processing via microfluidics. A not yet implemented Ω-machine design (Goranovic et al., 2006) is discussed both as a complementation mechanism (life-support system) for the development of protocells and as a potentially digitally controlled interface to program (evolve) protocellular functionalities.

[1]M. DeClue, P-A. Monnard, J. Bailey, J. Boncella, S. Rasmussen & H. Ziock, Information molecule mediated photocatalysis of container formation in protocells, submitted, 2007. [2] G. Goranovic, S. Rasmussen & P. Nielsen, Artificial life-forms in microfluidic computers, *in Proceedings, mTAS2006. 10th International Conf. on Miniaturized Systems for Chemistry and Life Sciences, Tokyo, Japan, November 5-9, 2006. [3]* Munteneau, Rasmussen, Ziock, and Sole, Generic Darwinian selection in catalytic protocell assemblies Emergence of protocellular growth laws, *Phil Trans. R. Soc. B*, in press, 2007 [4] PACE: European Commission sponsored 6th Framework Project "Programmable Artificial Cell Evolution" based on the Ω-machine concept, which was coined by Bedau, McCaskill, Packard & Ramussen, in Cannobio, 2002. [5]Rasmussen, Chen, Deamer, Krakauer, Packard, Stadler & Bedau, Transitions between nonliving and living matter, *Science* **303** (2004) 963 [6] Rouchelau, Rasmussen, Nielsen, Jacobi & Ziock, Emergence of protocellular growth laws, *Phil TransR. Soc. B*, in press, 2007.

Biomolecular Automata Using Deoxyribozymes: Accomplishments and Open Problems

Darko Stefanovic

University of New Mexico

Deoxyribozymes are nucleic acid enzymes which catalyze DNA reactions and which are largely made out of DNA. In our case, the enzymes, called phosphodiesterases, catalyze the reaction of cleavage of an oligonucleotide substrate (short sequence of single-stranded DNA) into two shorter oligonucleotides. Also in our case, the enzymes are modified to include allosteric regulation sites to which specific control molecules can bind and so affect the catalytic activity. The enzyme is modularly designed to include a type of regulation site to which a control molecule must bind before the enzyme can complex with the substrate, thus the control molecule promotes catalytic activity; another type of regulation site allows the control molecule to alter the conformation of the enzyme's catalytic core, such that even if the substrate has bound to the enzyme, no cleavage occurs; thus this control molecule suppresses catalytic activity. We call the allosterically regulated enzyme a logic gate, we call the control molecules the inputs to the logic gate, and we call the cleavage products the outputs of the logic gate. This basic logic gate corresponds to a multi-input conjunction, possibly with various input polarities. These are the first chemical logic gates in which inputs and outputs are of the same kind (namely oligonucleotides), so cascading gates is possible without any external interfaces (such as e.g., photoelectronics). The primary inputs are compatible with sensor molecules that could detect small molecules, or generally, cellular disease markers. Final outputs can be tied to release of small molecules, in particular drug molecules. Thus, it will eventually be possible to make therapeutic decisions cell-by-cell according to a complex decision function based on many attributes of the cell. This is sometimes referred to as ``intelligent drug delivery". On the one hand, the limits of this approach to intelligent drug delivery are defined by the extent of clinical knowledge about what kind of decision making is needed; on the other, the limits depend on what decision functions are feasible. In this talk, I shall explore the limits of deoxyribozyme logic devices from a computational standpoint, examining both what has been achieved in the wet lab and what is on the drawing board.

DNA Self-assembly and Computer System Fabrication

Chris Dwyer

Duke University

The migration of circuit fabrication technology from the microscale to the nanoscale has generated a great deal of interest in how the fundamental physical limitations of materials will change the way we engineer computer systems. The changing relationships between performance, defects, and cost have motivated research into so-called disruptive or exotic technologies. This talk will present the theory, design, and methods of fabrication for DNA self-assembled nanostructures within the context of circuit fabrication. The advantages of this technology go beyond the simple scaling of device feature sizes (sub-20nm) to enable new modes of computation that are impractical under the constraints of conventional fabrication methods. A brief survey of several computer architectures that can take advantage of this new technology will also be presented.

Quantum Computation: How Goes It?

Howard Barnum

LANL

I will survey the current state of quantum computation and the prospects and motivation for its further development. Specifically, I'll summarize the quantum model of computation, emphasizing that this provides a distinct theoretical model of computation from the classical one, and explainining the reasons for thinking it may be more powerful than the classical model but not sufficiently powerful to do NP-complete problems in polynomial time. This will provide concrete illustrations of the potential advantages of quantum computation for specific tasks, and an idea of the kinds of tasks quantum computation might be expected to speed up.

I'll then explain why it seems to be so hard to implement quantum computation, survey the state of the art in experimental physical implementation and the difficulties it faces, sketch how error correction and fault tolerant computation can in principle be implented for quantum computation in a manner closely analogous to the way they can be implemented for classical computation, and give some reasons for my subjective probability, of around 0.95, that quantum computers will be in use and outperforming classical computers on some tasks within 40 years.

Intrinsic Computation in Quantum Processes: An Information-Theoretic Analysis Using Quantum Finite-State Machines

Karoline Wiesner, John Mahoney, and James P. Crutchfield

University of California, Davis

We consider a quantum process as a quantum information source. Correlations in the generated sequences show that measured quantum systems store and process information in their behavior. We analyze this form of intrinsic computation by means of various information-theoretic quanti- ties: entropy rate, excess entropy and transient information. Using a quantum finite-state machine representation of a quantum system we analyze a set of small-scale quantum systems and find computational capacities of varying degree. The method exemplified here for small-scale system is generalizable to large-scale systems.

We recently introduced ways to measure information storage in quantum systems, using quantum finite-state machines (QFMs) as a computation-theoretic model that accounts for measurement effects [1,2]. We consider a quantum system, sub ject to a time-independent Hamiltonian and a measurement procedure, as a quantum information source. We refer to a quantum process as the joint probability distribution over a bi-infinite chain of variables produced by a quantum information source. A QFM is a representation of such a quantum process.

The information measure we introduced is the quantum entropy rate h_u, an equivalent to the classical entropy rate introduced by Shannon [3]. It measures the amount of true randomness in the process. For the class of so called deterministic QFMs we gave a closed-form expression for h_u in Ref. [2]. The next information measure, the quantum excess entropy, quantifies the shared information between a quantum process's past and its future. The third, the quantum transient information, determines the difficulty with which an observer comes to know the internal state of a quantum process through measurements. We use this set of measures to quantify the information storage and processing capacity or short, the computational capacity, of a quantum process. In this study, we analyze a set of quantum processes generated by QFMs with only a few states. We find a wide range of computational capacities for these small scale systems, depending on the Hamiltonian and the measurement procedure. The results encourage a new perspective on information storage and processing as being intrinsic to behavior of even simple quantum systems. The here presented methods are generalizable to quantum systems of any size. The approach presented here should lead to a new perspective on quantum processes and their computational capacities as they are observable in molecular systems. The importance of molecular systems as electronic memory has recently been illustrated [4]. Our information-theoretic analysis of dynamics is suggested as a guide in the search for new forms of computational substrates.

[1] K. Wiesner and J. P. Crutchfield. http://arxiv.org/ abs/quant-ph/0608206, 2006. [2] J. P. Crutchfield and K. Wiesner. http://arxiv.org/ abs/quant-ph/0611202, 2006. [3] C. E. Shannon. Bel l Sys. Tech. J., 27:379-423, 623-656, 1948. [4] J. E. Green and et al. Nature, 445:414-417, 2007.

Friday, March 23

08:30-09:10	**Seth Copen Goldstein** *Programming Programmable Matter*
09:10-09:50	**Jake Beal** *Properties for Engineered Emergence*
09:50-10:10	**Melanie Mitchell** *Four Principles of Adaptive Information Processing in Decentralized Systems*
10:10-10:40	Coffee
10:40-11:00	**Susan Stepney** *Robust, adaptable, powerful computation, as inspired by Nature: A Grand Challenge for Computing Research*
11:00-11:20	**Yoshihiko Horio and Kazuyuki Aihara** *Real-Number Computation through High-Dimensional Analog Physical Chaotic Neuro-Dynamics*
11:20-12:00	**Hava T. Siegelmann** *Analog State Space in Memories and in Stochastic Supra-Turing Computation*
12:00-02:00	Lunch *(on your own)*
02:00-02:40	**Jean-Louis Giavitto** *The Chemical Paradigm, Programming in Space and the Implementation of Autonomic Systems*
02:40-03:20	**Ann Bouchard** *Dynamic Self-Assembly as Computation: An Unconventional Approach to Software Implementation*
03:20-03:50	Coffee
03:50-04:30	**Fredric Gruau** *Programming self-developing "blob machines" for spatial computing.*
04:30-05:10	**Jonathan W. Mills** *Dr. Strangestuff or: How I Learned to Stop Programming and Love Plastic Foam*
05:10-05:15	Closing remarks

Programming Programmable Matter

Seth Copen Goldstein

Carnegie Mellon University

Taking the inevitability of (sub-)micron-scale manufacturing as a starting point, I will describe Claytronics, a new form of programmable matter composed of individual units designed to scale to millions of cooperating, sub-millimeter scale units. one of our driving goals is for the ensemble to act coherently and thereby mimic, with high-fidelity and in 3-dimensional solid form, the look, feel, and motion of macro-scale objects.

One the main challenges to realizing programmable matter is in programming the ensemble. The programmer must achieve a global goal for the ensemble by managing the collection of programs running on each unit in the ensemble. In addition to the normally hard problem of programming a distributed system, claytronics impose significant additional constraints, such as an ever changing topology of connections, limited resources, and uncertainty from operating in the physical world. In this talk we describe some of the challenges and proposed solutions for programming such ensembles.

Properties for Engineered Emergence

Jake Beal

MIT

It is difficult to establish engineering control over the behavior of aggregates of unreliable devices with complicated interaction patterns. I take a linguistic view of this problem, searching for mechanisms that simplify the composition and abstraction of complicated behaviors. From my work on various problems of aggregate control in cognitive architectures and spatial computing, I have noticed common themes in mechanisms that solve them. From these, I extract four properties which seem to help in engineering robust aggregate behavior---self-scaling, sparseness, gradual degradation, and failure simplification---and give examples of how they can be exploited.

Four Principles of Adaptive Information Processing in Decentralized Systems

Melanie Mitchell

Portland State University

Since the beginning of the computer age, the terms "information processing" and "computing" have been used to describe the dynamics of natural adaptive systems, ranging from the brain to cellular metabolism and genetic regulation. It is widely assumed that information processing in such systems takes place in a very different manner than in von Neumann-style computing architectures. In particular, the architectural features of these natural systems---large and varying numbers of relatively simple, stochastic, and noisy components, limited, dynamic, and unreliable connections, and no central control---require a radically different model of computation than the traditional one.

In this paper I describe the mechanisms underlying information processing in three different natural systems---the immune system, ant colonies, and cellular metabolism---and discuss the relevance of these mechanisms for adaptive behavior in these systems. I then abstract four general principles that I claim are key to adaptive information processing in decentralized systems such as these. These principles deal with the representation and transmission of information, the essential role of randomness, the importance of fine-grained parallel architectures, and the interplay of bottom-up and top-down processes in all such systems. Finally, I describe how these principles might be used in artificial intelligence or artificial life applications to achieve robust and fluid pattern recognition and learning.

Robust, adaptable, powerful computation, as inspired by Nature: A Grand Challenge for Computing Research

Susan Stepney

University of York, UK

How can we build complex computational systems – systems that are autonomous, adaptable, and robust – from millions of less reliable and simpler components? How can we build them to perform correctly and safely in an unpredictable, changing and hostile environment, and over long periods of time? Such tasks are currently well beyond the state of our computational art, and as our technology moves towards ever smaller and ever more numerous nano-scale and quantum devices, these tasks will get only more difficult.

And yet biological processes manage to do such things routinely. Living creatures are remarkably robust and adaptable. They can survive injury, damage, wear and tear, and continual attack from other creatures. Biology manages to take huge amounts of potentially unreliable matter and use self-checking, self-repair, self-reconfiguration, multiple levels of redundancy, multiple levels of defence, to develop adaptable complex biological organisms that continue to work for long periods of time in an extremely hostile environment.

So, in an attempt to cope with complexity, researchers are drawing inspiration from biology, which seems to have already discovered the answers, to develop a host of ***bio-inspired algorithms*** in evolution (genetic algorithms, genetic programming), neurology (artificial neural networks), immunology (artificial immune systems), plant growth (L-systems), social networks (ant colony optimisation), and more.

Researchers are also beginning to explore ***open complex adaptive systems***, where new resources, and new kinds of resources can be added at any time, either by external agency, or by the actions of the system itself. Such new resources can fundamentally alter the character of the system dynamics, and so allow new possibilities, new adaptations. Our current computational systems are beginning to open themselves, for example, through the continuing dialogue between user and machine, through continued new connections to networks such as the Internet, and through robotic systems controlling their own energy sources.

One of the most exciting, and seemingly weird, recent developments is the non-classical paradigm of ***quantum computing***. This has emphasised the fundamental link between computation and its physical embodiment in the real world. Still in relative infancy, it holds out the promise of massive increases in computation power, of untappable communication channels, and of spooky effects such as quantum teleportation.

I will describe the background and progress of one of the UK's Grand Challenges in Computing Research: "Journeys in Non-Classical Computation" [1], which seeks to uncover, explore, and generalise all the many diverse non-classical computational paradigms, to produce **a fully mature and rich science of all forms of computation, that unifies the classical and non-classical (natural) computational paradigms**. Such a mature computational science will allow us to design and build robust, adaptable, powerful, safe, complex computational systems. It will help researchers to uncover deep biological truths: which features of biology are necessary for correct robust functioning (so true of any living organism)? Which are necessary only because of the particular physical embodiment (carbon-based terrestrial life-forms)? Which are merely contingent evolutionary aspects (and so could be different if "the tape were played again")? It will help researchers to uncover deep physical truths: Which are more fundamental, the laws of computation, or the laws of physics? What is the relationship between logical information (bits) and physical reality? Are there corresponding laws of computation for the various "non-elementary" physical laws (for example, solid state physics)?

[1] S. Stepney, S. L. Braunstein, J. A. Clark, A. Tyrrell, A. Adamatzky, R. E. Smith, T. Addis, C. Johnson, J. Timmis, P. Welch, R. Milner, D. Partridge. Journeys in Non-Classical Computation I: A Grand Challenge for computing research. *Int. J. Parallel, Emergent and Distributed Systems* **20**(1):5-19, March 2005; and II: Initial journeys and waypoints. *Int. J. Parallel, Emergent and Distributed Systems*. **21**(2):97-125, April 2006

Real-Number Computation through High-Dimensional Analog Physical Chaotic Neuro-Dynamics

Yoshihiko Horio and Kazuyuki Aihara

Tokyo Denki University

Conventional von Neumann computers have difficulty in solving many complex, large-scale, and ill-posed real-world problems, which include combinatorial optimization problems. However, living things often face such a problem in real-life, and should quickly obtain a suitable answer to it through physical, dynamical, and collective computation with a vast number of neuron assemblies. This highly-parallel computation through a high-dimensional dynamics (Computation through Dynamics) is completely different from a numerical computation of the von Neumann computers (Computation through Algorithm).

A chaotic itinerancy is a universal dynamics in high-dimensional chaotic systems. It is observed also from activities of living things, in particular, from the brain activities. Therefore, it is highly probable that chaotic itinerancy dynamics is useful, effective, and important in information processing for living things. The complexity of the chaotic dynamics comes from the complexity of real-number. However, conventional digital computers cannot handle almost all real numbers. Consequently, we must use a physical (real-number) system such as analog electrical circuits with continuous variables to fully exploit the computational ability of the chaotic systems.

In this paper, we explore a novel computational mechanism based on the real-number processing with high-dimensional physical chaotic dynamics. We first develop two hardware systems using analog chaotic neuron integrated circuits, which can handle real-number through their continuous state variables, in order to solve quadratic assignment problems (QAPs) with chaotic itinerancy dynamics. These systems are designed considering the hierarchy of information processing in the brain. The first system takes a top-down approach. That is, a solution of the QAP is constructed by observing an 800-dimensional chaotic dynamics of the analog chaotic neural network. In the second system with a bottom-up approach, a heuristic algorithm, the Tabu search in our case, is driven by a chaotic neuro-dynamics with 300 dimensions. Through experiments, we demonstrate that both systems efficiently solve the QAPs through the physical chaotic dynamics. Moreover, we analyze underlying mechanism of the highly parallel and collective analog computation from a nonlinear dynamics point of view. Furthermore, we are currently extending our hardware systems for large-size QAPs which a conventional digital computer cannot solve.

Analog State Space in Memories and in Stochastic Supra-Turing Computation

Hava Siegelman

University of Massachusetts, Amherst

In many dynamical systems that model memory, discrete attractors are formed in a relatively small number of locations in the state space. In biological memory systems, however, there seem to be a continuum of attractors. Also current models show how a single input triggers a well specified attractor, while in biological systems it takes a stream of inputs for form a flow in memory space. We study Input-Driven Dynamic Attractors which are affected by both input dynamics and the trade-off between input dynamics and inherent network dynamics. We demonstrate that because the dynamics are affected by continuously valued parameters a continuum of attractors emerge. The flow in attractor space is dependent not only on the last inputs but on the complete history, it may be complex and bifurcation emerge, and the attractors themselves can be more complex than fixed points. Our model has the ability to dynamically explain Attention biases, Memory formation and reconsolidation, as well as Context effects.

We continue to explain why when continuous state space is involved the computation surpasses that of the classical digital models of computation. We also demonstrate a similar effect for stochastic or asynchronous discrete models which sample from external analog processes.

The Chemical Paradigm, Programming in Space and the Implementation of Autonomic Systems

Jean-Louis Giavitto

CNRS & Université d'Evry Val d'Essone

The chemical reaction metaphor (Banâtre, Le Metayer 1993) describes computation in terms of a chemical solution ı which molecules (representing data) interact freely according to reaction rules. Chemical solutions are represented y multisets (data-structure that allows several occurrences of the same element). Computation proceeds by ewritings which consume and produce new elements according to conditions and transformation rules. The result of a hemical program is obtained when a stable state is reached i.e., when no reaction can take place anymore. For nstance, the reaction:

primes = **replace** x, y **by** y **if** x div y

omputes the prime numbers lower or equal to a given number N when applied to the multiset of all numbers between 2 and N. The reaction replaces any couple of elements x and y such that the reaction condition holds (x div y is true if and only if x divises y). This process goes on until a stable state is reached, that is to say, when no divisible element emains.

The goal of Autonomic Computing is to realize computer and software systems that can manage themselves using self-organization, self-healing and self-optimization properties. The chemical paradigm has been advocated as well suited to express autonomic properties: the reaction rules correspond to the local actions to be taken to react to a perturbation.

We believe that the relevance of the chemical paradigm for the specification and the high-level programming of large autonomic and parallel/distributed systems comes from two fundamental characteristics:

1. the multiset data structure and the multiset rewriting device which suitably represent the orderless interactions (reactions) between elements that occur in large parallel or open systems;
2. the computation of a stable state such that self-healing, self-protection and self-optimization behaviors can been seen as the stabilization of the system on a fixed point after a transient perturbation (in this point of view, an chemical program is a dynamical system: a state is a multiset, a trajectory of the dynamical system is a program run and a stable state is the result of the program run).

However, these two general statements must be refined:

1. The direct interactions of arbitrary elements in a system are not always allowed nor desirable. The system may exhibit some data organization and only "neighbor" elements may interact. The neighborhood relationship may represent physical (spatial distribution, localization of the resources) or logical constraints (inherent to the problem to be solved).
2. A multiset stable w.r.t. the reactions represents a solution computed by the program or an admissible state of an autonomic system. This state is best characterized by *global* properties (e.g. the *entire* multiset of primes lower or equal to a given number) while the reactions represents *local* changes (e.g. the suppression of *one* integer fulfilling some conditions). Therefore, the *real difficulty* of chemical programming lies in the linking of the local changes to the desired global property.

In this presentation, we will present some concepts and tools in the field of algebraic topology that can be used to face these two problems.

As showed in the MGS project (Giavitto, Michel; 2002) (Giavitto 2003) topology can be used to constrain the multiset structure to more adequate organization. Indeed, constraining the universal topology provided by multisets to nested sequences leads to Lindenmayer systems (Rozenberg, Saloma, 1992); restriction to discrete lattice corresponds to cellular automata and more general crystalline computations (Margolus, 1999). And topology with higher dimension can be used to give a direct finite formulation to field equations of classical physical theories (Tonti, 1972) with obvious interests for the numerical applications (Palmer, Shapiro 1994; Tonti, 2001).

The formalization of such topological structure relies on the notion of chain complex (Munkres, 1984). A chain complex is a cellular space (a discrete space build by gluing elementary spaces or atomic localizations) and labelled by values. Chain complex can be used to formalize a data structure, its organization and its rewriting (Giavitto, Michel, 2002). As a consequence, a chemical reaction (on a multiset or on a more constrained structure) is a function from chain to chain. Such transformations can be very arbitrary, especially when the pattern language is sophisticated. However, it appears that, in a lot of applications, the reactions are restricted to chain homomorphisms (reactions that respect the algebraic structure of chains). These chain homomorphisms are also called *cochains*. The computational content of a cochain is clear; it corresponds to a simple pattern of distributed computations and hence can be viewed as a skeletons that package useful and reusable patterns of parallel computations in the line of the work (Skillicorn, 1996). Interestingly, cochains can be interpreted as *discrete differential forms* (Desbrun et al., 2006; Spicher 2006). Intuitively, a differential form is an entity that appear under the integral sign: it is a local function applied on each "point" of a domain to obtain a global result. Various theorems have been developed in mathematical analysis to link the local (differential) and the global (domain) level: existence of a solution of a differential equation, Stockes theorem, integration by part, etc. Each of these results can be translated into the discrete setting and provide tools to reason on the global properties emerging of the local reactions.

[Banâtre, Le Métayer, 1993] Jean-Pierre Banâtre and Daniel Le Métayer. Programming by multiset transformation. Communications of the ACM (CACM), 36(1):98–111 (January 1993). [Giavitto, Michel; 2002] Jean-Louis Giavitto and Olivier Michel. Data Structure as Topological Spaces. In Proceedings of the Unconventional Models of Computation: Third International Conference (UMC 2002), LNCS 2509, Springer 2002. [Giavitto; 2003] Jean-Louis Giavitto. Topological Collections, Transformations and Their Application to the Modelling and the Simulation of Dynamical Systems. In Proceedings of 14th International Conference on Rewriting Techniques and Applications (RTA 2003), LNCS 2706, pages 208–233, Springer 2003. [Rozenberg, Saloma, 1992] G. Rozenberg and A. Salomaa. Lindenmayer Systems: Impacts on Theoretical Computer Science, Computer Graphics and Developmental Biology, Springer Verlag, 1992. [Margolus, 1999] Norman H. Margolus. Crystalline Computation. In Feynman and computation: exploring the limits of computers, pages 267–305, Perseus Books, 1999. [Tonti, 1972] Enzo Tonti. On the mathematical structure of a large class of physical theories. Accademia Nazionale dei Lincei, estratto dai Rendiconti della Classe di Scienze fisiche, matematiche e naturali, Serie VIII, Vol. LII, fasc. 1, Janvier 1972. [Tonti, 2001] Enzo Tonti. A Direct Discrete Formulation of Field Laws: The Cell Method. CMES - Computer Modeling in Engineering & Sciences, vol.2, no.2, 2001, pages 237–258. [Palmer, Shapiro 1994] R. S. Palmer, V, Shapiro. Chain models of physical behavior for engineering analysis and design. Research in Engineering Design, Vol.5, No. 3, 1994, [Giavitto et al., 2005] Jean-Louis Giavitto, Olivier Michel, Julien Cohen, Antoine Spicher, Computation in Space and Space in Computation. In Unconventional Programming Paradigms (UPP'04), LNCS 3566, page 137–152, Springer 2005 [Bird, de Moor, 1997] Richard Bird and Oege de Moor. Algebra of programming. Prentice Hall, international series in computer science, 1997. [Munkres, 1984] James Munkres. Elements of Algebraic Topology. Addison-Wesley, 1984. [Giavitto, Michel, 2002] J.-L. Giavitto and O. Michel. The topological structures of membrane computing. Fundamenta Informaticae, 49:107–129, 2002. [Skillicorn, 1996] David B. Skillicorn. Communication skeletons. In Abstract Machine Models for Parallel and Distributed Computing, pages 163–178, IOS Press, The Netherlands, 1996. [Desbrun et al., 2006] Mathieu Desbrun, Eva Kanso, and Yiying Tong. Discrete differential forms for computational modeling. In Discrete differential geometry: an applied introduction, pages 39–54. SIGGRAPH'06 course notes, ACM, 2006. [Spicher 2006] A. Spicher. Transformation de collections topologiques de dimension arbitraire - Application à la modélisation des systèmes dynamiques. PhD Thesis. Université d'Évry-Val d'Essonne, décembre 2006.

Dynamic Self-Assembly as Computation: An Unconventional Approach to Software Implementation

Ann M. Bouchard and Gordon C. Osbourn

Sandia

We have shown that protein dynamic self-assembly processes map naturally onto a formal model of computing, the random access machine (RAM). Specifically, we have shown that many biological processes that do not appear to be information processing, such as synthesis of macromolecules, structural assembly, transport, disassembly, and degradation of molecules, actually do perform computation when viewed from the RAM perspective [1].

Dynamic self-assembly brings several novel properties to computational systems. In biological systems, algorithms yield physical, functional structures, i.e., structures that perform some function, or act as a machine. The assembly of the structure from its constituent parts automatically enables the structure to perform its function,Äîactuate, signal, build other structures, or tear them down. These structures are not fixed or rigid, but instead are easily altered, temporarily combined, and even disassembled, in response to changing conditions or signals. This provides pathways for information and material flow that can be created on-the-fly where needed, and dismantled to terminate these flows. This malleability is not found in many conventional hardware and software architectures.

The inherent flexibility and reconfigurability of these processes inspired us to base a novel software technology on dynamic self-assembly. The fundamental software entity, termed a "machine," is designed to embody the dynamic self-assembly properties of proteins. Machines have dynamic binding sites that can selectively bind to sites of other machines. The binding or unbinding of sites can trigger data passing, execution, or the exposing, hiding or altering of other sites. The change of other sites can in turn trigger subsequent binding or unbinding of those sites, resulting in further data passing, execution, etc.

The net result is the dynamic formation of functional software assemblies (a collection of bound machines) that provide data and execution pathways. The software functionality at any give time is determined by the current structure of the machine assemblies. This in turn is determined by the self-assembly and disassembly processes. A pathway can be reconfigured (machines unbound, and possibly re-bound to other machines), and this reconfiguration (effectively, assembly of a different pathway) results in different functionality.

In this talk, we will demonstrate the software functionality of self-assembling pathways using a graphical user interface (GUI) we have implemented with this approach. For a series of examples, we will describe the user interaction with the GUI (e.g., typing, mouse movement, left click), the pathway self-assembly that is triggered by that user event, and the functionality resulting from the assembly of that pathway. Some novel capabilities that are enabled by the inherently dynamic nature of the self-assembling executable code will also be discussed.

[1] A. M. Bouchard and G. C. Osbourn ,"Dynamic self-assembly in living systems as computation", Natural Computing, 5 (4), 321-362, 2006.

Programming self-developing "blob machines" for spatial computing.

Fredric Gruau

University of Paris Sud

We present blob computing: A Blob is a generic primitive used to structure a uniform computing substrate into an easier-to-program parallel virtual machine. We find inherent limitations in the main trend of today,Äôs parallel computing, and propose an alternative model trying to combine both scalability and programmability. In order to allow scalability of hardware, we consider "spatial computing media" such as 2D cellular automata, amorphous computers, FPGA, or 2D grids of processors. In order to achieve the programmability, we program this hardware using two levels:

The first "system level": It is a run time system implemented on the computing medium. It takes the form of a local rule which can maintain connected regions called blobs. Blobs can be encapsulated. A blob is similar to a deformable elastic membrane filled with a gas of atoms. (elementary empty blobs). Blobs are interconnected using channels, which act as springs and bring connected blobs closer to each other. The system implements in a distributed way: the movement, the duplication and the deletion of blobs and channels. It can also propagate successive waves to communicate signals in a pipelined way: intra-blob, or inter-blob.

The second "programmable level": The run time system installs a virtual machine called "blob machine" above the computing medium. A blob is programmed using a finite state automaton, with output actions triggering duplication, deletion, movement or communication . Execution starts with a unique ancestor blob that repeatedly duplicates and creates new blobs and new channels, thus generating a network of automata. This virtual blob machine is an example of "self-developing machines", which is easier to be programmed than a computing medium, or a VLSI circuit, because of its ability to dynamically create or delete new nodes.

We present the blob machine, how to implement blobs, and how one can program with blobs by using a higher level language called "blob ml". We illustrate the execution of many examples of small programs. They all have an optimal complexity, in the VLSI model of complexity, under some reasonable hypothesis, concerning the -not yet fully implemented- system level.

Dr. Strangestuff or: How I Learned to Stop Programming and Love Plastic Foam

Jonathan W. Mills

Indiana University

After ten years of experience with an increasingly sophisticated succession of Rubel's Extended Analog Computers (EACs), some general insights into this new paradigm of computing have emerged. They answer questions such as "How is it possible to compute with machines ranging from deviceless silicon ('empty space') to plastic foam to Jell-O(r) brand gelatin?", "Why does analogy compute efficiently?", "How are applications developed for these machines?" and "How do unconventional computers compare to digital computers-- and why should we use them?"

This session explains the difference between algorithmic digital computation and computation based on the direct use of the laws of physics and the mathematical principles that are implicitly and inherently present in matter. Applications of the EAC that illustrate this paradigm will be presented. Some have been prototyped at Indiana University, others are under evaluation in industry, and still others point toward the revolutionary devices that may be realized in the next ten years.

Demonstrations of the current generation of EAC will be available during the workshop.

Posters

Eleonora Bilotta and Pietro Pantano
Modelling the growth of artificial organisms by using 2D Self-replicating CAs

Paul Bogdan
In Network Computation through On-Chip Stochastic Communication

Rene Doursat
Embryomorphic Engineering: How to Design Hyper-Distributed Architectures Capable of Autonomous Segmentation, Rescaling and Shaping

Harold Fellermann, Steen Rasmussen, and Hans-Joachim Ziock
Self-replicating Nanocells as Possible Substrate for Chemical Computation

Luca Gammaitoni
Computing with noise takes time

Philip H. Goodman, Rene Doursat, Quan Zou, Milind Zirpe, and Oscar Sessions
RAIN Brains: Mammalian Neocortex as a Hybrid Analog-Digital Computer

Ralph Hollingsworth
Spatial Design Using Binding P-Systems

Christian Jacob
Spatial Swarms: A New Computing Paradigm?

Simon D. Levy
Continuous States and Distributed Symbols: Toward a Biological Theory of Computation

Genaro Juarez Martinez and Harold V. McIntosh and Andrew Adamatzky and Juan C. Seck Tuoh Mora
Designing collision-based computers in elementary cellular automata

Nebu John Mathai and Takis Zourntos
A Hierarchical Dynamical System Analog Computation Architecture for Embodied Cognition

Elebeoba E. May
Bits and Bases, Parallel Paradigms for Communication

Reid Porter and Garrett Kenyon
Fault Tolerance via Local Adaptation

Brian F. Rossa
An analogical definition of computation

Gabriele Scheler
Computing with cortical microcircuits---new suggestions for parallel programming models

Hideaki Suzuki
Towards self-organized computation with a network

Moise Valvassori
Components and Aspects Oriented Language for Amorphous Computers

Modelling the growth of artificial organisms by using 2D Self-replicating CAs

Eleonora Bilotta and Pietro Pantano
Università della Calabria, Italy

Starting from the metaphor that self-reproducers have life-like properties and that they are organisms (or agents) that live in an environment, 2D self-replicating cellular automata (CAs) -- found by evolutionary techniques -- provide insight into analogous processes in the biological world, computing in a highly unconventional way.

We have identified many self-reproducing methods. In this talk we present three of these mechanisms and a new way of managing information, useful for developing new forms of computation. The first mechanism (**Universal Constructor self-reproducing method)** is very similar to stem cell behavior since it is possible to reproduce any configuration given as input, according to the way in which one organizes spatially the structure one wishes to reproduce. In particular, for many of these self-reproducer systems it is necessary that these structures have the same spatial orientation, or that they be turned at 180° and it is necessary that they be settled at a distance of almost two cells, or a multiple number of 4 cells. In turn, these rules can be combined together in order to produce second order dynamics, in a further complexification of the self-replicated structures. The second mechanism (**Developmental self-reproducing method**) is organized in two distinct phases. At the early stage, the structures become bigger and after relevant steps of simulation reproduce a copy of themselves. In this stage, the self-reproducers grow as biological organisms do and give life to child copies of themselves, which in turn grow and procreate after many steps of simulation. Offspring develop in the opposite direction to the parent. Meanwhile, the parent continues to grow, gradually changing its configuration. In other words, the replication of developmental self-reproducers involves several nested processes, operating on different time scales. The main process is driven by the original self-reproducers and their coordinated production of offspring. Each of these offspring gives rise, in turn, to new asynchronous processes. These are emergent and are not explicitly encoded in the original agents.

The third mechanism is the method we have called **ad hoc mixed systems,** which allows us to explore the potential of creating 2D self-reproducing CA patterns at will. Ad hoc mixed system modules are independent mathematical entities, which can themselves be represented by Boolean CAs. In some cases, the module overlaps with the self-reproducing initial data; in others, the initial data can be recovered by combining different modules. In our work, we implemented a computational system which is able to identify modules in the patterns generated by a self-reproducer, (i.e., the basic self-reproducer or a subset of the basic self-reproducer). This made it possible to simulate the 2D CA using an ECA rule. In other words, the final pattern is obtained by combining subsets of the same pattern. Each of these substructures is governed by an ECA rule that defines its development, on different scales.

All the self-reproducer systems presented above use genetic rules to guide the building of their final configurations in the same way that the biological genetic code governs the production of proteins. To extend this metaphor and to create a new way of computing, we mapped the state of one system into the language of DNA bases, associating 0 with the code A (adenine), 1 with G (guanine), 2 with T (thymine), 3 with C (cytosine) and 4 with U (uracil). This enabled us to count the numbers of each "base" present at different stages in the growth process, obtaining dynamical networks which allow for the comprehension of the system's codification, and the maintenance, sending and transcription of this information.

The relevance of such work consists the possibility of achieving the goal of modeling the growth of artificial systems with the possibility of developing these methods in new physical and computational devices, especially devoted to nanotechnologies and DNA computing.

In Network Computation through On-Chip Stochastic Communication

Paul Bogdan
Carnegie Mellon University

Shrinking feature sizes, increasing interconnect and logic density, and continuous scaling of the supply voltage and clock speed make the future System-on-Chips (SoCs) highly sensitive to neutron and alpha radiation and susceptible to a large number of timing violations. Consequently, the manufacturability of VLSI chips in the deep-submicron (DSM) domain requires new design and verification methodologies. The unpredictable failure of such VLSI chips calls for the relaxation of correctness standards for physical interconnections and the use of alternative communication paradigms. Thus, the Network-on-Chip (NoC), which is a promising solution to communication problems faced in the SoC context, needs to support fault tolerant communication. Our work aims to investigate the theoretical foundations of a biologically inspired communication approach. More specifically, we propose an analytical model for the on-chip stochastic communication paradigm. We utilize our model to analyze the number of nodes aware of the packet dissemination versus time and validate our conclusions with simulation results.

Embryomorphic Engineering: How to Design Hyper-Distributed Architectures Capable of Autonomous Segmentation, Rescaling and Shaping

René Doursat
CNRS & Ecole Polytechnique, France

Exploding growth in hardware components, software modules and network users forces us to find an alternative to rigidly designing computational systems in every detail. Future progress in information and communication technologies will depend on our ability to, instead, "meta-design" mechanisms allowing those systems to self-assemble, self-regulate and evolve. Such decentralized, autonomous systems are already pervasive in nature and called "complex", although they are often less costly, more efficient and even simpler than intelligently designed, centralized systems. Complex systems are characterized by the self-organization of a great number of small, repeated elements into large-scale patterns, where each element may itself obey the dynamics of an inner network of smaller entities at a finer scale (microprogram). The new engineering challenge is to "guide" this self-organization, i.e., prepare the conditions and mechanisms favorable to a robust and reproducible—as opposed to random—pattern formation process (macroprogram). Yet, at the same time, it is also to let the parameters of this process evolve in order to freely generate innovative designs. Finding efficient systems requires matching loose selection criteria with productive variation mechanisms. The first point concerns the openness of the designers to "surprising" outcomes; the second point concerns the intrinsic ability of complex systems to create a "solution-rich" space by combinatorial tinkering on highly redundant parts. Embryogenesis, the development of an entire organism from a single cell, provides the most striking example of self-organization guided by evolvable genetic information. This work presents an original model of artificial *embryomorphic* system growth. A virtual organism is represented by a mass of cells that proliferate, migrate and self-pattern into differentiated domains. Each cell contains an internal gene regulatory network and acquires specific gene expression identity by interaction with neighboring cells. Different identities trigger different cell behaviors, which in turn induce new identities. The final organism's architecture depends on the detailed interplay between the various rates of cell division and movement, propagation of genetic expression and positional information. Ultimately, on this score of "theme and variations" (developmental laws and parameters), evolution will be the player. In possible hardware applications of this model, nano-units containing the same instructions could self-organize without the need for reliability or precise arrangement as in traditional VLSI. In software or network applications (servers, security, etc.), a swarm of small-footprint software agents could diversify and self-deploy to achieve a desired level of functionality. In all cases, embryomorphic architectures suggest a "fine-grain" approach to systems design, i.e., one based on hyper-distributed collectives of a great number of very simple and relatively ignorant cloned elements.

Self-replicating Nanocells as Possible Substrate for Chemical Computation

Harold Fellermann, Steen Rasmussen, and Hans-Joachim Ziock
LANL

This work presents the design of an artificial organism [1] that is currently being pursued both in vitro and in silico in an international, interdisciplinary research project [2]. In our approach, a micellar or vesicular proto-container is catalytically coupled to a proto-genome and a light-driven metabolic reaction. Although inspired by nowadays single cellular organisms, the approach differs significantly in its actual chemical implementation and can be understood as a minimal implementation of a chemical system that possess features attributed to living systems, namely self-assembly, growth, self-replication, and the possibility to evolve.

Here, a mesoscopic computer model of the minimal protocell is presented [3,4] in which molecules are represented in a coarse grained fashion and their 3D motion is calculated by the dissipative particle dynamics (DPD) formalism [5,6]. Additionally, the model incorporates chemical reactions by means of stochastic process. Simulation results of the self-assembly, the growth by incorporation and metabolic turnover of precursor molecules, the template directed replication of the genome, and the fission of the aggregate into two separate daughter cells are presented.

We outline a possible concept to use protocellular biofilms as a substrate for chemical based computation, in that protocells can maintain internal states by means of hydrophobic reactants that undergo multistable reactions within the cells, while information of states can be propagated among cells by diffusion of hydrophilic messengers. Logical gates are achieved by stoichiometric coupling between these messengers. We hope to increase the complexity of these gates by the use of genetic biopolymers either as messenger or catalyst. Furthermore, we can confine the metabolic activity of the protocells to arbitrary topologies by shining light patterns on the artificial biofilm. This allows wiring individual gates to more complex logical networks. While this design adopts prominent approaches in chemical computation [7-9], the protocellular system presented in this work additionally offers the features of self-replication and evolution. We are currently seeking for computational concepts that harvest these features.

[1] Rasmussen et al, Artificial Life 9:269-316, 2003. [2] Los Alamos National Laboratory sponsored Protocell Assembly (PAs) project and the European Commission sponsored Programmable Artificial Cell Evolution (PACE) project. 2004-08 [3] Fellermann & Sole, Phil. Trans. R. Soc. B., (in press 2007) [4] Fellermann et al., Artificial Life, submitted 2006. [5] Hoogerbrugge & Koelman, Europhys. Lett. 19:155-160, 1992. [6] Groot & Warren, J. Chem. Phys. 107(11):4423-4435, 1997. [7] Hjelmfelt, Weinberger & Ross, PNAS 88(24):10983-10987 [8] Kuhnert, Agladze, & Krinsky, Nature 337:224-247, 1989 [9] Seelig et al., Science 314:1585-1588, 2006

Computing with noise takes time

Luca Gammaitoni
Universita' di Perugia, Italy

The present tendency to scale down CMOS based devices toward the nano-meter region1 makes the discussion of the role of noise in computation increasingly relevant. Noise immunity in a low energy dissipation scenario has become the recurring objective of significant research efforts in this field. On the other hand some authors have focused their attention on the potential role of noise in nanoscale devices where noise driven dynamics has been invoked to explain the experiments and to optimize future design. In order to address a non-negligible error probability a number of strategies have been devised where a probabilistic approach to the computational task has been often invoked. It is not our aim to propose a new computing strategy. Instead we demonstrate that the careful consideration of the noise effects, together with a proper choice of the ,Äúidle time interval,Äù lead to a prescription for the minimization of the error probability that can be summarized as follows: in a noisy environment you can still compute by using the usual threshold-crossing logic gates, provided that you wait long enough, but not too long. We anticipate this result to be relevant toward the design and realization of nano-scale computer where ambient noise, instead of being a mere source of disturbances could be an essential component of the computing process itself.

RAIN Brains: Mammalian Neocortex as a Hybrid Analog-Digital Computer

Philip H. Goodman[1], Rene Doursat[1,2], Quan Zou[1], Milind Zirpe[1], and Oscar Sessions[1]
[1]University of Nevada, Reno. [2]CNRS & Ecole Polytechnique, Paris, France

What kind of computer is the mammalian brain? To improve upon simple rate-based artificial neural networks, computational neuroscience research over the past decade focused on more biologically realistic spiking neuron models—but still ascribing, on the millisecond time scale, a *digital* overtone to brain processing. A more recent development has been to explore the spectral properties of subthreshold membrane potentials, emphasizing an *analog* mode of computing. Together, by modeling the fine temporal structure of neural signals, these research trends have revealed a great diversity of collective *spatiotemporal regimes*: synchronization and phase locking, delayed correlations and traveling waves, rhythms and chaos, etc. Through recurrent (and plastic) synaptic connections, neural cells transiently interact as *dynamical subnetworks* that promise an immense richness of coding expression and computational power, combining the discrete and the continuous. What repertoire of dynamical regimes ("phase diagrams") can such subnetworks sustain? In the classical feedforward view, subnetworks (layers, cell assemblies) are initially mostly silent and have to be literally activated by an input or a "lower" area. Our work subscribes to a new paradigm, in which subnetworks already possess viable and complex endogenous activity modes that are only *perturbed* through coupling with an input or other subnetworks. Using spiking neuronal simulations, we describe here progress to-date towards building cohesive "analog-digital perturbation" principles that can underlie biological attention, pattern recognition, short- and long-term memory, and motor responsiveness to natural environmental stimuli. In particular, we describe the performance and sensitivity of dynamically igniting-and-quenching *R*ecurrent *A*synchronous *I*rregular *N*etworks (RAINs). We explore the regimes and phase transitions of RAINs under conditions of calibrated voltage-sensitive ionic membrane channels, synaptic facilitation and depression, and Hebbian spike-timing dependent plasticity (STDP). Specifically, we demonstrate the spontaneous emergence of alternating sub-100 millisecond states of subthreshold (i.e., analog) correlation-decorrelation, suggesting a natural mechanism of intrinsic clocking. We also study "lock and key" properties of RAIN activation, i.e., a model of pattern recognition and nondiscrete memory storage based on a dynamics of *coherence induction* triggered by input stimuli (the "keys"). Here, learning a pattern (a "lock") means tuning synaptic efficacies to a point of maximal postsynaptic response. Finally, we discuss the importance of embodied social robotics to "teach" intelligent behavior to RAIN brains, and speculate on the instantiation of RAIN brains in compact analog VLSI architectures.

Spatial Design Using Binding P-Systems

Ralph Hollingsworth
Muskingum College

Numerous computing technologies are under current or proposed development, in fields ranging from biology to classical electronics. A novel aspect of much of this work is the dynamic nature of the computing devices, i.e. the ability to extensively reconfigure physical structures during computation. To shift classic solution designs into such devices, languages used to implement solutions will need to be cognizant of topological and geometric factors. In order to study issues related to such "programming" languages, a conceptual framework has been developed. This environment extends P-Systems1 to include inter-object binding. Initial prototype studies have used PyMOL2 and Python to create examples of docking among membrane structures found in P-Systems. By using the classes in PyMOL, originally designed for molecular studies, it is possible to create abstract objects that can attract/repel each other via mutual docking sites. The membrane character of P-Systems provides a methodology for enclosure, and for the exchange of parameter information with buffering (trans-membrane transport). Enclosed structures may be further composed, while sheltered from outside environments. The current study includes the external binding of membrane objects to each other, and implements their binding using geometric/topological hashing. Such hashes provide a significant heuristic capability for reducing the intractability found in complex, 3-dimensional docking. As a goal, a language structure based on these ideas could be applied to computing devices, including fixed-matrix substrates, chemical assemblages, and biological cells, and could be used for scales ranging from nano-sized devices to macro-sized clusters of computing entities.

1. Paun, G., "Computing with Membranes", Journal of Computer and System Sciences, 61, 1 (2000), 108-143 2 .DeLano, W.L., "The PyMOL Molecular Graphics System (2002)", World Wide Web: http://www.pymol.org

Spatial Swarms: A New Computing Paradigm?

Christian Jacob
University of Calgary, Canada

The study of social interactions within human societies as well as insect colonies has led to a new field in artificial intelligence—the study of collective intelligence. The emergent properties of 'swarms of agents' are apparent in many facets. An ant colony, for example, exhibits properties of a hive mind: the colony grows, takes in food, has to get rid of its waste, has to defend its foraging space against other colonies—short, it displays many properties of a living entity. However, the colony only exists through the network of its more or less tightly connected building blocks, the ants.

Particularly interesting from a computing perspective is the fact that no matter on which scale we observe a collectively intelligent system, we can utilize the same or very similar principles to capture a system's emergent properties. Consequently, we can reuse algorithms for traffic systems [1,9] to model army ant raiding patterns [11]. Algorithms that describe the interaction of immune system cells [6,7] can as well be used for modeling how gene regulatory processes work inside a bacterial cell [2,3,4,10]. We have even applied swarm computing principles in art exhibitions with interactive computer installations that provide playful exposures to swarms that create art and music [5].

Swarm systems like these perform massively parallel computations in a spatial environment, which can not be captured well enough through relatively static cellular automata. My lab is exploring new ways to model, understand, and utilize swarm-based systems. We have recently introduced Swarm Grammars [8,12] (as an extension of Lindenmayer systems) to control thousands of swarm agents interacting in 3D space through rule-based systems. The rule systems can be evolved, so that we breed a co-evolutionary ecology of swarm computations, which are represented as 3-dimensional, structural designs—thus creating computational landscapes, similar to the hive minds of social insects.

[1] R. Hoar, J. Penner, and C. Jacob. Evolutionary swarm traffic: If ant roads had traffic lights. In IEEE World Congress on Computational Intelligence, Honolulu, Hawaii, 2002. IEEE Press. VirtualBacteria-Refs (2004) [2] R. Hoar, J. Penner, and C. Jacob. Transcription and evolution of a virtual bacteria culture. In IEEE Congress on Evolutionary Computation, Canberra, Australia, 2003. IEEE Press. [3] C. Jacob and I. Burleigh. Biomolecular swarms: An agent-based model of the lactose operon. Natural Computing, 3(4):361-ï376, December 2004. [4] C. Jacob and I. Burleigh. Genetic programming inside a cell. In T. Yu, R. L. Riolo, and B. Worzel, editors, Genetic Programming Theory and Practice III, pages 191‚Äì206. Springer Verlag, 2006. [5] C. Jacob, G. Hushlak, J. Boyd, P. Nuytten, M. Sayles, and M. Pilat. Swarmart: Interactive art from swarm intelligence. Leonardo (in print), 40(3), 2007 [6] C. Jacob, J. Litorco, and L. Lee. Immunity through swarms: Agent-based simulations of the human immune system. In Artificial Immune Systems, ICARIS 2004, Third International Conference, Catania, Italy, 2004. LNCS 3239, Springer. [7] C. Jacob, S. Steil, and K. Bergmann. The swarming body: Simulating the decentralized defenses of immunity. In Artificial Immune Systems, ICARIS 2006, 5th International Conference, Oeiras, Portugal, September 2006. Springer. [8] C. Jacob and S. von Mammen. Swarm grammars: growing dynamic structures in 3d agent spaces. Digital Creativity, 18(1), 2007. [9] J. Penner, R. Hoar, and C. Jacob. Swarm-based traffic simulation with evolutionary traffic light adaptation. In L. Ubertini, editor, Applied Simulation and Modelling, Crete, Greece, 2002. ACTA Press, Zurich. [10] J. Penner, R. Hoar, and C. Jacob. Bacterial chemotaxis in silico. In ACAL 2003, First Australian Conference on Artificial Life, Canberra, Australia, 2003. [11] G. Suen. Modelling and simulating army ant raids. M.sc. thesis, University of Calgary, Dept. of Computer Science, Calgary, Canada, April 2004. [12] S. von Mammen, C. Jacob, and G. Kokai. Evolving swarms that build 3d structures. In CEC 2005, IEEE Congress on Evolutionary Computation, Edinburgh, UK, 2005. IEEE Press.

Continuous States and Distributed Symbols: Toward a Biological Theory of Computation

Simon D. Levy
Washington & Lee University

The classical theory of computation rests on two fundamental assumptions: states are finite, and symbols are atomic. Although automata built on these assumptions are extremely successful at solving many computational tasks, the assumptions are highly implausible for human and animal cognition. First, the signals used by the brain and other biological systems are mainly continuous, as evidenced by the widespread use of differential equations in modeling these systems. For this reason, it makes little sense to view mental states as countable, let alone finite. Second, there is very little reason to believe that mental representations involve locally-stored atomic symbols. Consequently, classical pointer-based discrete structures over such symbols, and algorithms operating on such structures, are not biologically realistic. Experimental evidence instead favors a view in which the representations of entities, concepts, relations, etc., are distributed over a large number of individually meaningless elements in a way that supports similarity metrics and content-based retrieval.

Although both continuous-state computation and distributed representations have received a fair amount of research attention, it is uncommon to see them discussed together in the unconventional-computation literature (except, perhaps, as part of a general survey). In our presentation we argue that a biologically plausible theory of computation will require both a continuous-state automaton component and a distributed-memory component, much as a classical pushdown automaton uses both a finite-state automaton and a pushdown stack. This view is supported by current research in clinical psychiatry suggesting hemispheric differentiation for sequence processing and conceptual structure.

We show further that stack-like operations (PUSH and POP) over distributed representations can be performed as simple vector addition and scalar multiplication, in a way reminiscent of moiré or foreground/background effects in visual processing. This possibility suggests that "higher" mental functions like language and abstract thought might be exploiting existing neural circuitry already available for other purposes. We conclude with a simple application example from language parsing, and some speculation about possible new directions and guiding principles for biologically-inspired unconventional computation.

Designing collision-based computers in elementary cellular automata

Genaro Juarez Martinez[1], Harold V. McIntosh Andrew Adamatzky[1] and Juan C. Seck Tuoh Mora
[1]*University of the West of England*

A collision-based computer is an architecture-less device, where signals are represented by particles (gliders or self-localizations), and logical operations are calculated by collisions among particles. In this work we present results about detecting and classifying all possible collisions between particles in one-dimensional cellular automata, and constructing a catalogue of binary collisions. Using this catalogue, we develop formal languages via de Bruijn diagrams to represent particles in the evolution space of Rule 54 and Rule 110 cellular automata using regular expressions. Finally, we describe a subset of regular expression to code each known particle in Rule 54 and Rule 110 to solve some relevant problems like self-organization in particle dynamics, production of glider guns and logical operations.

A Hierarchical Dynamical System Analog Computation Architecture for Embodied Cognition

Nebu John Mathai and Takis Zourntos
Texas A&M University

We present an unconventional analog computation paradigm based on the application of cybernetics to the synthesis of cognitive faculties for mobile robots with life-like behavior. We are encouraged by a long history of ideas linking life, cognition and computation. Sommerhoff, formulating the basis for an analytical biology, sought to explain the apparent purposiveness of living systems. Cyberneticists sought to identify and exploit the control and communication mechanisms underlying natural and artificial autonomous goal-directed systems, believing regulation to apply as much to an organism's externally directed cognition as to the homeostatic control of its internal environment. Maturana and Varela (echoing Wiener) very directly state, "living systems are cognitive systems and living is a process of cognition." Finally, recent work on the "artificial life route to artificial intelligence" seeks to enrich AI---and, by extension, the underlying computational mechanisms that facilitate cognition---through the development of embodied robotic agents with life-like structure and behavior.

Our approach is cybernetic and influenced by artificial life insights. We view the computational machinery underlying cognition to be regulatory and hence use control theoretic synthesis tools. Moreover, we develop physically embodied cognition and utilize a hierarchical dynamical systems structure. Our architecture consists of a cascade of regulators separated by time scale, each having access to sensor data and which control problems (with varying levels of abstraction) relating to the agent's motion in the world. To synthesize the regulators we employ a methodology of plant-controller co-design; after constructing a plant that models the salient environmental dynamics, we apply nonlinear control theoretic toolsets to synthesize a corresponding controller. In addition to providing a systematic means of developing analog computation systems, we find that the use of the feedback regulator motif as a computational primitive naturally facilitates parallelism.

Conceptually our approach has affinities with Ashby's notion of intelligence amplification, i.e., we design a cognitive dynamical system coupled to a "problem" via sensor/actuator channels, and whose stability yields a solution in a more complex problem domain. Indeed, we highlight the emergence of satisficing intelligence to demonstrate this, appealing to Bedau's definition of weak emergence. As well, we show how satisficing intelligence arises as a result of our unconventional approach to stability---we relax the pursuit of asymptotic stability to ultimate boundedness, giving the system flexibility to consider more varied actions. Finally, simulation results and animations (available at http://www.ece.tamu.edu/~takis/robotics_main_page.html) for the system in non-trivial environment topologies are provided, illustrating the life-like behavior of the cognitive system and the emergence of satisficing intelligence.

Bits and Bases, Parallel Paradigms for Communication

Elebeoba E. May
Sandia

Constructing parallels between engineering frameworks for communication and biological information processing mechanisms is a challenge that if embraced can contribute to our quantitative understanding of complex molecular processes. Forming a quantitative understanding of how biological organisms can communicate their genetic message efficiently in the presence of noise can improve and advance current communication protocols. Given this objective, the underlying hypothesis of the work presented is that the genetic system can be operationally paralleled to an engineering communication system, which transmits and operates on bases as opposed to bits [May, et al., 2004]. The central dogma of genetics can be viewed as the paradigm for biological communication, where an organism's redundancy containing DNA is the result of an error control encoding process. We use this framework to study several problems in molecular biology: 1) The informational capacity of the genetic replication process; 2) Aging related mutagenesis; and 3) Detection of single nucleotide polymorphisms in DNA sequences.

Mutations are replication errors that remain or are missed by genetic proofreading mechanisms. Mutation derived capacity values can provide insight regarding the information capacity of the replication process. We calculate the genetic channel capacity using u_b, the single base mutation rates reported in Drake et al. [Drake et al., 1998]. Assuming the DNA replication process can be paralleled to a discrete memoryless channel, we analyze the information capacity of replication using empirical mutagenesis data and the Shannon entropy to calculate the channel capacity for the following prokaryotic and eukaryotic systems: Bacteriophage M13, Bacteriophage Lambda, Bacteriophages T2 and T4, Escherichia coli, Saccharomyces cerevisiae, and Neurospora crassa), higher eukaryotes (Caenorhabditis elegans, Drosophila melanogaster, Mus musculus (mouse), Homo sapiens). We found that prokaryotic organisms have larger channel capacity values, ranging from 1.95 to 1.975 bits, than the higher eukaryotes with capacity values ranging from 0.4 to 1.85 bits. This suggests that for DNA microbes the maximum coding rate R is closer to (n-1)/n, leaving few bases for error control coding. In contrast, the channel capacity values for higher eukaryotes imply a distinctly smaller maximum value for R, suggesting that eukaryotic genomes have more bases available for error control coding, i.e. more redundancy in their genome. It is generally accepted that eukaryotic organisms have a greater number of "extra" bases in their genome (bases that are not used to specify amino acids) than prokaryotes; our findings are consistent with this belief.

Given the probability of reduced replication fidelity as the number of times a chromosome is copied increases, and the existence of biological mechanisms that prevent the continued transmission of error containing chromosomes, it may be possible to view aging and related mutation engendered diseases as inevitable communication failures.

Extending the genome replication channel model, it is evident that for a fixed u_b <1, as G (genome size) increases, the quantity (1-u_b)^G decreases. Consequently the channel error probability (1-(1-u_b^G) increases. The result is a reduction in overall channel capacity. We evaluate the probability of error for an organism's replication channel after NCD cell divisions by equating multiple cell divisions to the transmission of a genome of size G*NCD, where NCD is the number of cell divisions. There is a reduction in the information capacity of the replication channel for N_{CD}=1...75 cellular generations for higher eukaryotic organisms, substantiating the need for error control within DNA in order to ensure the survivability of an organism and ultimately the species.

Hybridization-based target recognition and discrimination is central not only to in vivo biological processes, but also vital to the operation of nucleic acid sensor systems. Therefore developing algorithms to accurately detect polymorphisms in DNA sequences is critical. Using the genetic communication theory paradigm, we investigate coding theory algorithms for uniquely categorizing single nucleotide polymorphisms in silico based on the calculation of syndromes. We develop a syndrome generation algorithm that can be used to uniquely distinguish SNPs in the middle to 3' end of the 15-base probe sequence of computational catalytic molecular beacons (deoxyribozyme gates) [Stojanovic and Stefanovic, 2003]. This error control coding algorithm not only detects a mutation in the sequence but can categorize the location and type of base mutation based on the unique syndrome vectors. We continue to investigate a complementary algorithm that produces similar results for SNPs in the 5' end of the input sequence and the biological implications of our communication theory algorithms.

Acknowledgment: SNP work performed in collaboration with S. Brozik, P. Dolan, and P. Crozier of Sandia National Laboratory. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company for the United States Department of Energy's National Nuclear Security Administration under contract DEAC0494AL85000. References JW Drake, B Charlesworth, D Charlesworth, JF Crow. Genetics, 148: 1667-1686, 1998. EE May, MA Vouk, DL Bitzer, DI Rosnick. BioSystems, 2004. MN Stojanovic, D Stefanovic. Nature Biotechnology, 21: 1069-107, 2003.

Fault Tolerance via Local Adaptation

Reid Porter and Garrett Kenyon
LANL

For applications like image and signal processing and real-time control, artificial neural networks, due to their distributed cellular architecture, can provide natural solutions for computing with fault prone devices. The neural network community has proposed several architectures like Hopfield Networks and Self Organizing Maps that exploit system dynamics to solve computational problems. Solutions are represented as equilibrium points and the convergence of the system effectively provides self-organized fault tolerance. This type of approach appears ideal for fault tolerant computing with next generation devices but it currently limited to specific applications for which these systems have been proposed e.g. associative memories and clustering. In this poster we investigate how fault tolerant equilibrium can be used in applications which are not explicitly encoded in the system dynamics. The basic approach is very simple and amounts to relaxing the requirements. Instead of requiring the system dynamics to directly encode the desired computation, we require only that the system dynamics include the desired computation as a local equilibrium. We then use the initial conditions to bootstrap the system to the desired local equilibrium and hence exploit fault tolerant equilibrium for a much wider class of application.

An analogical definition of computation

Brian F. Rossa
Lockheed Martin Advanced Technology Laboratories

The word "computation" is overloaded. Its semantics lend application to countless situations, both spontaneous and contrived, natural and artificial. Our intuition tells us what computation is and what it isn't, but the scientific vernacular often colludes with what can be said, formally, about mathematical constructs, physical systems, and the act of computing. This work proposes a rigorous qualitative definition of computation that hopes to resolve these ambiguities. We define computation as a process of analogy-making between physical systems and give evidence to support that this is a particularly good definition. We go on to describe classic results in computer science in terms of this new model, disambiguating it from other definitions of computation. We show that this definition succeeds in unifying much thinking about the relationship between computation and physics.

"Computation" is evoked notionally in a variety contexts. So many, in fact, that the naive investigator cannot say with much certainty what exactly computation "is." In the academic computer science community, "computation" is associated almost exclusively with the Turing Machine, and historically with mathematics. If a problem cannot be solved with a Turing Machine, then it's said to be "uncomputable." Yet we find that recently this association has been made less salient by those who are interested in analog computation. In most contexts, "computation" implies a computer, a physical system (or perhaps a "physically realizable" system) whose states encode the entities of interest. Yet the formal connection between abstract computation and its physical realization is not addressed outside the work of a small group of prescient physicists. Finally, it is difficult to imagine "computation" as anything but a human endeavor, and yet we find the entire field of "natural computation" as a counterpoint to this intuition.

The nascent study of computation across all such contexts - let us call it "computation science" to avoid further overloading – cannot come to fruition without resolving such ambiguities. We argue that the development of a computation science depends upon a rigorous, unified definition of computation. This work presents a body of evidence - albeit circumstantial evidence – to support our argument. When taken in the whole, this evidence admits a very strong candidate:

We find that computation, in the most general sense, is about the making of a very particular analogy between two physical systems. These systems we will call the "source" and the "computer" for obvious reasons. Once constructed, this analogy between source and computer is simple and precise. It says that the states of the computer, evolving according to its own dynamics, encode what "could" happen to the source system. In order for us to say that a physical process is an instance of computation in this sense, it must satisfy certain criteria. First, the source and computer must be distinct systems. Second, we must identify a process whereby observables of the source system are mapped to controls on the computer. This process we call "embedding." Embedding, we claim, requires a third distinct physical system that is as important to computation as the source or computer. This system we call the "constructor."

The notion of a constructor is implicit in previous attempts to define computation in physical terms. Toffoli [2005] argues very nicely that computation does not exist outside a very particular context. Many others have described computation as something of a goal-directed process. Latent in all of this is a physical system - usually a human- that embodies goals, establishes context, and ultimately serves as the constructor in our sense. We choose to make this system's role explicit because, we will show, it serves a very important “physical” role in computation. Examples will be given shortly of many different instances of computation that support our intuition.

We choose not to define computation in terms of symbols or abstract machines or languages, but rather in terms of physical systems. Our analogical definition of computation (ADC), then, is a “physical” definition of computation. This is motivated by a great deal of contemporary research in “nonstandard computation” that allows our intuition about what computation “is” to grow beyond the digital model. We are also motivated by a number of very difficult problems in computer engineering that remain unresolved, presumably, because previous definitions failed to make explicit the relationship between computation and its physical substrate. In this endeavor, we draw from a wonderful body of work, starting with Landauer, that is concerned with the physical aspects of computation. Finally, we hope to collect under our banner a number of curiosities having to do with the computing of physics (rather than the physics of computation). These, we believe, offer a strange paradox against which to test the fitness of our ADC and, ultimately, pose questions about the so-called Church-Turing Deutsch Thesis.

Computing with cortical microcircuits- new suggestions for parallel programming models

Gabriele Scheler
Stanford University

I propose a computational model of cortical micro-circuitry consisting of the following parts: a) local circuits which compute by aligning and convoluting input signals with stored patterns using parametrization for afferent and recurrent filters (data stream) b) linking local circuits as coupled oscillators in global landscapes using interaction by synchronized signals (command level). I will focus here on computation in the local microcircuit.

System layout. Each local circuit has an arrangement of processors with afferent filters which are sigmoidals parametrized by *m*0 (and upper bounds (*u*), lower bounds(*l*)) and *s* or *k* and recurrent filters which express temporal modulation (p-modulation) of a sigmoidal function.

$f(x):=l+u/(1+e^{-s(1+(x-m0)^{k})})$ "afferent filter"

$f(p(x)):=l+u/(1+e^{-s(1+(x-m0(p(t))^{k})})$ and

$p(t):=1-1/(e^{t}+e^{(-pt)})$ "recurrent filter"

Afferent signals consist of systematic (statistically dependent) variations of a time-varying signal. The use of sigmoidals retains peak information across all filters and has the capacity to transform signals into a series of impulses. Clusters of *m*0 variation perform horizontal segmentation of the family of time-varying signals, and clusters of slope (*s*,*k*) variation perform the matching of its signal shapes to stored templates. The recurrent connectivity is matched to the afferent filter properties of each unit and is specific to each local circuit. The recurrent filter system adds up signal components according to the topology of connections and re-inserts signals to processors of the local circuit through p-modulated sigmoidal filters. P-modulation allows parametrized temporal dislocation of peaks in the input signals.

Computation: The system can be tuned such that prominent matches of signal shape between input and stored template generate strong synchronized signals (which are distributed to other local modules). This provides parallel processing by global connectivity on the command level, resulting in phase reset, phase advance or phase delay signals. On the data stream, the output is a combination of the components of the input signals and the signals that are stored by the parametrization of the processors (the local memory). This means that local circuits work both as acceptors (comparators) of input and stored signal, and as computational processors using input and stored signals. Storage or memory in the system consists of parameters as programmed elements vs. auto-tuned parameters necessary for calibration. We present a number of integer computations which can be realized with the system. Learning algorithms would allow programming by slow and selective adaption to input strings (not implemented). We also raise the question of (resource-restricted) universality vs. limitations of integer computation dependent on choice of filters.

Towards Self-organized Computation with a Network

Hideaki Suzuki
National Institute of Information and Communications Technology

Network Artificial Chemistry (NAC) [1,2,3,4] is a research approach that represents molecular interactions purely using a mathematical graph with no coordinate information. Molecular activities such as collisions and reactions are governed by spatial constraints in the real chemistry, so in order for the NAC to tightly emulate bio-molecular activities in the water, a rewiring rule of the NAC should be able to reproduce a regular structure in the graph through the rewiring rule.

From this notion, this paper proposes preparing an active program in the NAC node, and tests the possibility of creating regular structures. We take the two-dimensional square lattice as an example, and implement a program in each node. The algorithm extends two paths from a node, and if they do not meet and create a length-four loop, newly creates an edge for the loop. All the nodes in the graph has the same program, and through their parallel and independent execution, a quasi-regular structure is created.

Figure 1 shows the results obtained from a typical simulation run. We start from an initial random graph with 512 nodes and uniform degree 4. After conducting one hundred active operations for each node, a regular structure emerges in the graph, and the graph seems a kind of a folded sheet of a 2D square lattice (Fig.1(a)). As time goes on, the regular structure is kept basically, but the sheet structure is unraveled gradually, and after one thousand operations, we have the half-sheet and half-string topology shown in Fig.1(b).

In the previous studies of the NAC, we classified the types of edges from (weakest) van der Waals to (strongest) covalent and assumed that the weakest edges are basically rewired by a passive rewiring rule incorporating the physical/spatial constraint [2,3]. Solvent nodes were assumed to be inert (inactive) and the active operations were only realized by a data-flow cluster created through the mathematical folding of an active node chain [4]. The present results, however, suggest that when we try to create organized structure in a network, one of the easiest and straightforward way is to implement an active program in a node that makes the nodes behave in a complicated manner to some extent.

Conclusion: We designed an active program that locally rewires the surrounding edges of a solvent node in the NAC, conducted the numerical experiments of its execution, and succeeded in organizing a global regular structure in the NAC graph.

Future extention of this study includes improving the program so it might organize a lump structure, putting heterogenious programs in the NAC nodes to organize more complex structure, and so on.

This study was supported in part by Doshisha University's Research Promotion Funds, "Academic Frontier, Intelligent Information Science", in Japan.

[1]Suzuki, H.: Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems (Artificial Life IX) (2004) 507-513 [2] Suzuki, H., Ono, N.: The 8th European Conference on Artificial Life (ECAL) Workshop Proceedings CD-ROM. Canterbury, UK (2005) [3]Suzuki, H.: Australian Journal of Chemistry 59 (2006) 869-873 Suzuki, H.: BioSystems 87 (2007) 125-135

Components and Aspects Oriented Language for Amorphous Computers

Moise Valvassori
Paris 8 University, France

We design a component and aspect oriented language for amorphous computers[1]. A "software component" is piece of software which exposes predefined aspects and able to use others components. An aspect [2] is a part of a program that cross-cuts components. Components allow us to build libraries and aspects permit to build multiple backends such as simulations or embedded devices programs. Our amorphous computers consist of several thousands of cells distributed uniformly on a square surface. Each cell contains: a unique program p for all cells, n private states, n' public states called messages, and a blackboard b containing the neighborhood messages.

In our language, a program is composed by a components collection. A component contains a name and an aspects list in: parameters declaration, a components list used by this component (each component could be parameterized), a states list which represent the private state of the cell, a messages list which will be automatically transmitted to the cells in the neighborhood when modified, initialization programs, messages handling programs and general programs.

The previously listed aspects are expressed in a simplified Lisp dialect. Additionally to traditional functions and programming structures, five specific cell functions are available: a pair of functions for reading and modifying the private state of the cell, a pair of functions for reading and modifying the public state of the cell, a function for reading of the public state in the blackboard of the close cells, a access function to the parameters.

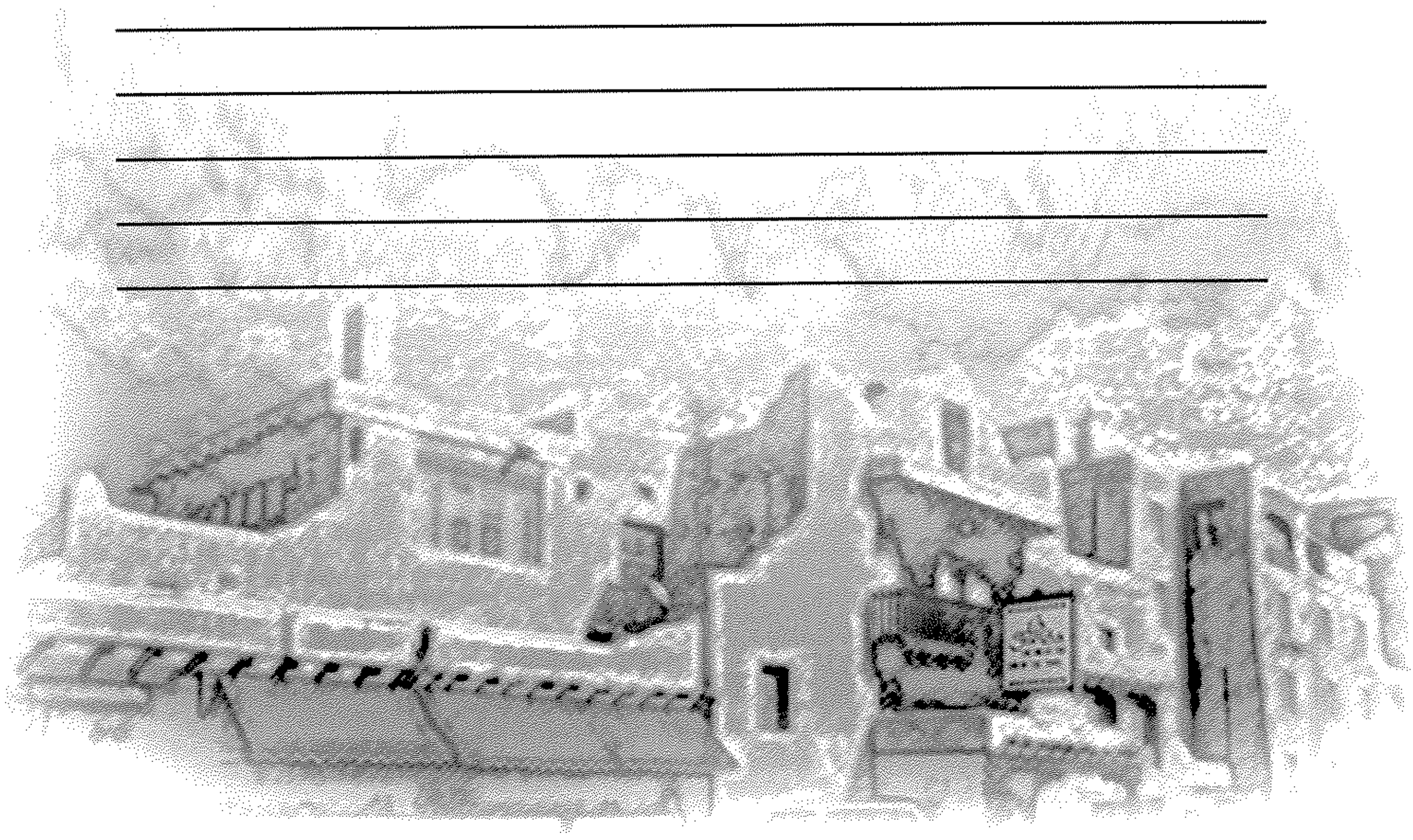
The components constituting an amorphous program are compiled to an autonomous simulator written in the C programming language.

Simulations created by the compiler simulate an amorphous computer involving several tens of thousands to several million cells. It's designed around a simple and efficient architecture: a table contains the complete state of all the cells, a table contains the neighborhood of all the cells, a set of threads dedicated to messages management, a set of threads dedicated to cell's behavior management, a thread specialized to data analysis, a thread coordinating the other simulation threads.

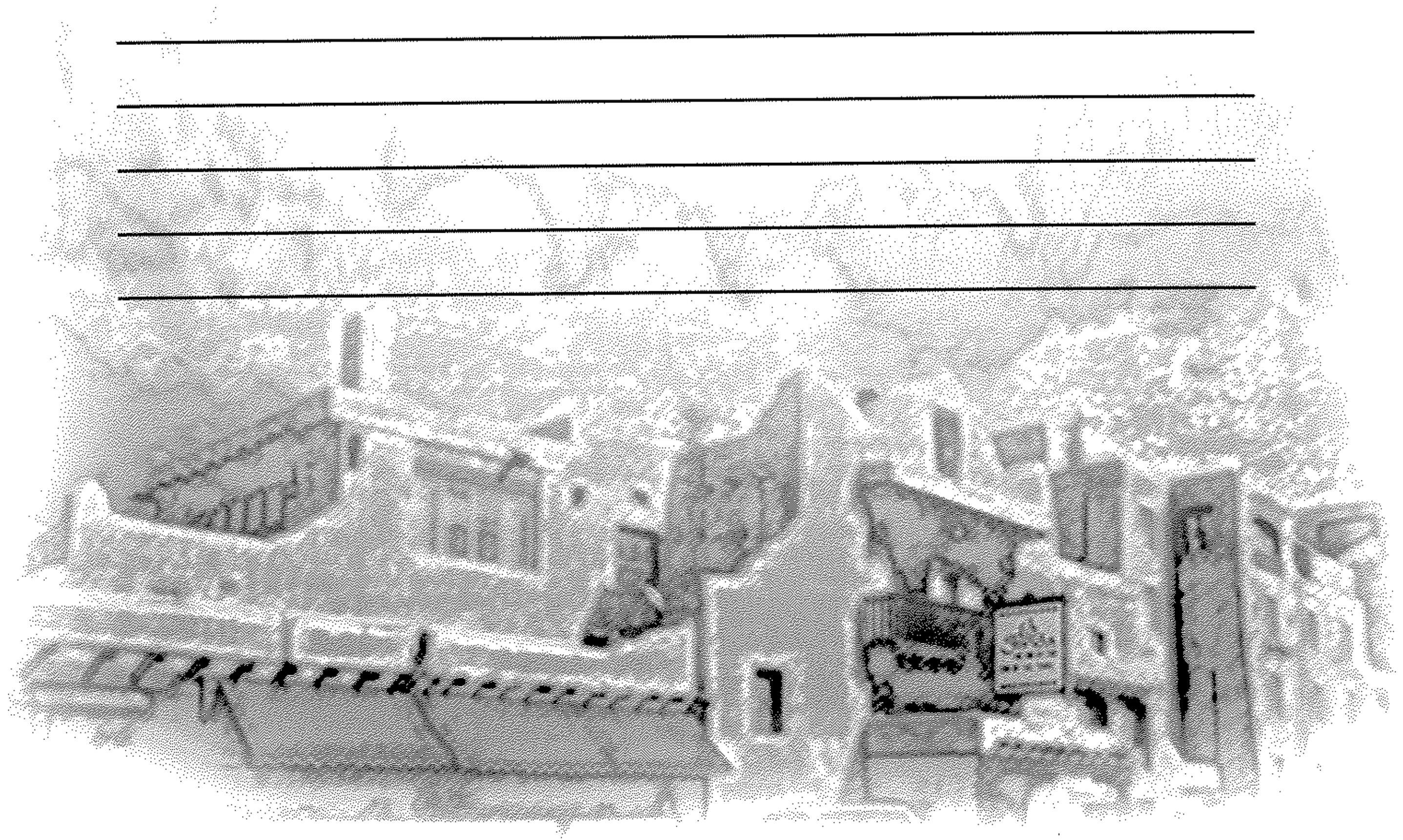
The figure 1 shows a Voronoi diagram component source code. This component uses another component which propagate the source cell's ids to other cells. Cells detecting id collision are on a partition's frontier. The figure 2 shows a visualization of a Voronoi diagram on an amorphous computer.

[1] H. Abelson et al. Amorphous Computing. CACM, 43(5):74-82, may 2000. [2] G. Kiczales et al. Aspect-Oriented Programming. In ECOOP'97,1997.

Notes:



Notes:



Notes: