Grand Challenge 7 : Journeys in Non-Classical Computation

Susan Stepney [1], Samson Abramsky [2], Andy Adamatzky [3], Colin Johnson [4], Jon Timmis [1,5]

[1] Department of Computer Science, University of York
[2] Computing Laboratory, University of Oxford
[3] Department of Computer Science, University of the West of England
[4] Computing Laboratory, University of Kent
[5] Department of Electronics, University of York
[1] *susan@cs.york.ac.uk*

We review progress in Grand Challenge 7 : Journeys in Non-Classical Computation. We overview GC7-related events, review some background work in certain aspects of GC7 (hypercomputation, bio-inspired computation, and embodied computation) and identify some of the unifying challenges. We review the progress in implementations of one class of non-classical computers: reaction-diffusion systems. We conclude with warnings about "regression to the classical".

Grand challenges, hypercomputation, bio-inspired computing, embodied computing, reaction-diffusion computers

1. INTRODUCTION

In 2002, the UK Computing Research Committee (UKCRC) issued a call for "grand challenges in Computer Science", to help focus and direct research. More than one hundred responses were received, and over a 2 day workshop these were amalgamated and boiled down into seven distinct Grand Challenges. GC7, *Journeys in Non-Classical Computation*, is one of those original seven. Subsequently some challenges have been merged, and others suggested; see the Grand Challenges website for current details[1,2].

The GC7 Challenge is *to produce a fully mature science of all forms of computation, that unifies the classical and non-classical paradigms*. The non-classical paradigms include computation under non-classical laws of physics (quantum theory, general relativity), biological computing, embodied computing, *in materio* computing, and more. Details of the Challenge, and some suggested research directions, can be found in [Stepney et al 2005a] [Stepney et al 2006a]. Here we report on some of the progress made in specific challenge areas.

Caveat: not all the research discussed here has been explicitly done as part of GC7, and some of the authors referenced may well never even have heard of GC7. Our purpose here is simply to survey some of the background work that we wish to synthesise into the overall vision of GC7. We start by listing some recent GC7-related activities (§2), then discuss some challenges in more detail (§3), and some implementations of non-classical computers (§4). We conclude with a warning against comfort, and a reminder of the goals of GC7 (§5).

2. ACTIVITIES SUMMARY

Several conferences and workshops have been organised around GC7, or in collaboration with GC7, or have featured GC7 discussions or presentations (see referenced websites and publications for more details):

- The Grand Challenge in Non-Classical Computation International Workshop (18-19 April 2005, York UK)[3] [IJUC 2006] [IJUC 2007] [Alexander 2009]
- The 5th International Conference on Unconventional Computation (4-8 September 2006, York UK)[4] [Calude et al 2006] [NCJ 2008] [IJUC 2008b] with the associated one day Workshop "From Utopian to Genuine Unconventional Computers" [Adamatzky & Teuscher 2006] [IJUC 2008a]
- International Interdisciplinary Workshop on Future Trends in Hypercomputation (11-13 September 2006, Sheffield UK)[5] [IJUC 2009]
- Unconventional Computation: Quo Vadis? (21-23 March 2007, Santa Fe NM USA)[6] [PhysD 2008]
- 2007 Conference on Unconventional Computing (12-14 July 2007, Bristol UK)[7] [Adamatzky et al 2007]

[1] http://www.ukcrc.org.uk/grand_challenges/
[2] http://www.cs.york.ac.uk/nature/gc7/
[3] http://www.cs.york.ac.uk/nature/workshop/ (sponsored by Microsoft Research Europe)
[4] http://www.cs.york.ac.uk/nature/uc06/ (part sponsored by EPRSC grant EP/D076420/1)
[5] http://www.hypercomputation.net/hypertrends06/workshop.html
[6] http://cnls.lanl.gov/uc07/
[7] http://uncomp.uwe.ac.uk/uc2007/ (part sponsored by EPRSC grant EP/E04381X/1)

- Thinking Through Computing workshop (2-3 November 2007, Warwick UK)[8]
- Automata 2008 (12-14 June 2008, Bristol UK)[9] [Adamatzky et al 2008] [JCA 2009]

Further events are planned, including a broader network of participants to work on GC7's overall goals.

3. CHALLENGES

GC7 is an umbrella activity. Its aim is to bring together individual journeys in non-classical computation that are being made by various researchers, both within and without the GC7 community, to ensure that a complete and coherent map of the territory is developed. We start here by reviewing activity in a variety of GC7 areas, and identifying the challenges they are exposing.

3.1 Hypercomputation

One way to challenge the underlying assumptions of the Turing model (for example [MacLennan 2004]) is in the details of its abstraction from the laws of physics. Investigation of this abstraction reveals that its underlying physical model is implicitly Newtonian.

If we instead consider the underlying physical laws to be quantum, we can have a situation where tape symbols can exist in superposition, can be entangled, and where quantum interference along different computational paths is crucial. Quantum computers allow for some quantum algorithms to exhibit exponential speedup over their most efficient classical counterparts, and for truly non-classical computations, such as genuine random numbers.

If we take general relativity into consideration, we can allow the Turing machine and the user who waits for it to halt to experience qualitatively different amounts of proper time (the user experiencing finite proper time whilst the Turing Machine experiences infinite proper time [Hogarth 1992]), such that the Halting Problem can (at least in principle) be solved. (Note that this approach subverts the classical proof of impossibility of the Halting Problem by requiring the computational device here to be a Turing Machine plus a space-time singularity: the program of this larger device cannot be encoded as a string that is fed back into itself; part of the implementation is physical.)

Considering the underlying laws of physics, as opposed to relying on the mathematical abstraction, can also outlaw certain challenges to the Turing model. For example, some researchers attempt to get "more than Turing" computation by challenging the assumption that the tape symbols are drawn from a finite set, and allowing real numbers into the computation. Real number analogue computers have (theoretically) more power than Turing machines. The reason why is easy to see: a single real number (given to infinite precision) contains an infinite amount of information. However, can such infinite precision quantities exist in the physical world? Can arbitrarily fine distinctions be made? At the very least, the enormous complexity of preparing such inputs, and measuring such outputs, needs to be taken it consideration [Blakey 2007]. But one also has to recognise that higher precision measurements require larger measuring devices and higher energies. Once the size of the measuring device reaches the size of the universe, no further precision is possible. Infinite precision real number computing is a physical impossibility.

One must be careful in arguing what is and is not physically possible. For example, consider accelerating Turing Machines [Copeland 2002]. The Turing model assumes an essentially constant time taken for each step in the computation. Challenging this, one can argue that an implementation that could do each step in half the time of the previous one could accomplish an infinite amount of computation in a finite time. Such classical accelerating Turing Machines can live in a Newtonian universe, but not in our relativistic one where the speed of light is a limiting factor, and infinite (or even unbounded) speed is a physical impossibility. However, general relativistic solutions, with the user and computer separated, such as mentioned above, that allow different proper times, have a similar effect.

In September 2006, an International Interdisciplinary Workshop on Future Trends in Hypercomputation was held at Sheffield UK, partly under the auspices of GC7. It was devoted to exploring many different ways to challenge the assumptions underlying the Turing Model. Several of the presentations were invited to be expanded for a special issue of the International Journal of Unconventional Computing [IJUC 2009], including one discussing hyper-computation from the perspective of GC7 [Stepney 2009]. Out of this workshop has grown the HyperNet: the Hypercomputation Research Network (EPSRC grant EP/E064183/1), run by Stannett.

Researchers have, of course, been speculating on the importance of physics in terms of limits on what a physical computer can do [Landauer 1961] [Lloyd 2000]. Some have speculated that computation is more fundamental than physics [Wheeler 1990]. The challenges include: how can new models of computation be abstracted from the underlying physics? How expressive can these models be? Are proposed models of computation implementable given the constraints of physics? Precisely what assumptions about the physical laws are being made? What other undiscovered assumptions underlie the Turing model?

[8] http://www.methodsnetwork.ac.uk/activities/act39.html
[9] http://uncomp.uwe.ac.uk/automata2008/

3.2 Interactive Computing

In some sense, the underlying model of physics is the simplest assumption to recognise and challenge, but there are others. For example, one can challenge the notion that the computation proceeds uninterrupted until it halts, at which point the answer is revealed. After all, biological organisms, often considered to be performing computation, do not "halt" (while they are alive, at least). [Wegner 1997] discusses interaction machines: "Turing machines extended by addition of input and output actions that support dynamic interaction with an external environment", and claims that they are more powerful than Turing machines. Whether or not this is the case, what we want to challenge here is the very notion of what it means to be "more powerful" than a Turing machine. If we play by the rules of the Turing paradigm, this is limited merely to considerations of halting, computability and computational complexity. These notions are certainly important, but so are other notions, including naturalness of models, of composition, of implementation, of proof, and so on. If an interactive approach is a more natural way of modelling or reasoning about a problem, then it is in some sense more powerful than a less amenable non-interactive model.

In this age of the Internet and the Web, and of pervasive and ubiquitous computing, *interaction* and *information flow* between *multiple agents* are key notions of computation. One approach to interaction is to cast the problem in terms of *games*, with two or more players interacting via strategies (where the environment can be cast as a player if required). For example, the 7th Augustus de Morgan Workshop (2005) was dedicated to this topic of "Interactive Logic, Games, and Social Software", with the manifesto:

> Traditionally, logic has dealt with the zero-agent notion of truth and the one-agent notion of reasoning. In the last decades, research focus in logic shifted from these topics to the vast field of "interactive logic", encompassing logics of communication and interaction. The main applications of this move to *n*-agent notions are logical approaches to games and social software. The wealth of applications in these areas will be the focus of the 7th Augustus de Morgan Workshop. [10]

[Abramsky 2006] discusses the logic and semantics of *n*-player games. Game semantics is yielding new approaches to compositional model-checking, and to analysis for programs with state, concurrency, probability and other features.

3.3 Bio-inspired Computing

The ultimate interactive computing is that performed by biological organisms. They are constantly interacting (exchanging matter, energy and information) with their environment, which may include other organisms. Viewing biological systems as performing computation, and taking inspiration from them to develop analogous algorithms, is possibly the most well-represented non-classical computation paradigm.

Computation today is rife with bio-inspired models (neural networks, evolutionary algorithms, artificial immune systems, swarm algorithms, social insect algorithms, and so on). However, many of these models are naïve and out of date with respect to the underlying biology. Without too much exaggeration, one could say that the area has been cherry-picked for over-simplistic ideas, which have been implemented into algorithms of dubious biological relevance. Initial attempts applied to small or very specialised problems were very successful (or were not published), but attempts for more general applicability, and for scalability, repeatedly hit snags.

The field is now maturing [Banzhaf et al 2006] [Timmis et al 2006]. Computer scientists are working closely with biologists to understand the full complexity of the systems they are attempting to emulate, and to work out principled abstractions that will allow properly engineered algorithms. As part of this process, we have suggested a conceptual framework for structuring the analysis of biological systems and the development of domain-specific engineered algorithms inspired by them [Stepney et al 2005b].

We have laid out a vision for how this would work in one particular domain, that of *Immuno-engineering* [Timmis et al 2008]. It can be argued that the immune system's speed, flexibility and multiple response options rely on a parallel-processing system which has "wasteful" use of resources, countless back-up systems, and requires the ability to immediately and continuously monitor physical sites. This has massive implications for the engineer, who is constrained by processing speeds, communication overheads, and physical resources, and furthermore hindered by hardware requirements such as transmitting signals from sensors, but who can freely make numerous copies of software agents, subject only to storage constraints. Immuno-engineering crucially takes into account these differences between artificial systems and biological systems: for example, the different numbers, kinds, and rates of signals that need to be monitored and processed; the different kinds of decisions that need to be made; the different effectors available to support and implement those decisions; and the different constraints of embodiment, either physically or virtually engineered. The Immuno-engineering approach takes steps to address these issues.

Computer scientists are seeking inspiration from all levels of biology, from DNA transcription and gene regulatory networks, metabolic networks, immune systems, endocrine systems, neural systems, slime moulds, plant growth, fungal mycelia, social insect behaviours, to populations and ecologies. As a small example, we are currently progressing the ideas in the field of evolutionary algorithms, by building more detailed models of bacterial genome

[10] http://www.illc.uva.nl/ADMW05/

structure and their evolutionary processes, in order to develop novel evolutionary algorithms with richer analogical computational structures, that are more suitable to solve complex dynamic control tasks [Stepney et al 2007], as part of the Plazzmid project (EPSRC grant EP/F031033/1).

The challenges here are grand indeed, not least because this work is progressing in parallel with the biological research itself. What are the underlying bio-computational mechanisms? What of these are contingent on the organism being an embodied carbon-based lifeform? Are there general principles that can be abstracted to other situations? How can this abstraction and mapping be performed when the application domain is radically different from the biological one? Can this work help the biological research programme?

3.4 Massive parallelism

One of the major features of biological systems is their massive parallelism, on multiple scales. Cells contain millions of molecules, organisms contain trillions of cells, ecosystems contain billions of organisms, all acting and interacting on different length- and timescales. In the Turing model, a parallel TM has no more computational power than a sequential one, since a parallel machine can be simulated by a sequential one. However, this argument applies only under the assumptions of the Turing model, which does not include considerations of real-time performance, naturalness of description, or fault tolerance and redundancy, among others.

One of the original areas discussed in the context of GC7 is concurrency [Stepney et al 2006a], where we argue that classical computer science has taken a wrong turn by focussing on sequential machines, and how concurrency, when approached correctly, should be the *natural* way to describe, model, and implement systems. We argue the case for process algebra-based languages, specifically, CSP [Hoare 1985] and π-calculus [Milner 1999]. (See also [Abramsky 2005] for a different take on some of the problems.) The TUNA feasibility project (EPRSC grant EP/C516966/1) investigated the use of these languages, and of an executable implementation in a *process-oriented* language (specifically occam-π [Welch & Barnes 2005]) for simulating the activity of platelets in blood clotting [Polack et al 2005] [Stepney et al 2006b] [Welch et al 2006]. This demonstrated that extremely efficient massively parallel simulations are readily derivable from process algebra descriptions. The work is continuing in the follow-on CoSMoS project ("Complex Systems Modelling and Simulation infrastructure", EPSRC grants EP/E053505/1, EP/E049419/1), which is a case-study driven approach to developing a method for modelling complex systems (case studies include immunological subsystems, plant ecologies, and liquid crystals) and deriving appropriate highly parallel process-oriented simulations [Andrews et al 2008] [Polack et al 2008].

The massive parallelism in TUNA/CoSMoS is achieved by timeslicing tens to hundreds of thousands of processes on a single processor, and distributing over several tens of processors (a rack of Linux boxes connected by Ethernet). The implementation language allows this distribution to be achieved relatively transparently. True concurrency on a single processor can be achieved on FPGAs (Field Programmable Gate Arrays), programmable hardware that allows different parts of a single chip to be executing different parts of a program at the same time. FPGAs can be linked together in large numbers (for example, the BioWall [Tempesti & Teuscher 2003]), allowing massive parallelism to be achieved relatively cheaply. The CSP-based language Handel-C can be used to program multiple FPGAs in a process-oriented style.

Such classical parallelism is relatively straightforward technologically, it merely requires a change in mindset to move from a "sequential-first" to a "concurrent-first" style. (Although this might be considered to be equivalent to saying that a more efficient keyboard is relatively straightforward technologically, it merely requires a change in mindset to move from a qwerty to a Dvorak style.) More challenging is the parallelism provided by many forms of analogue computing. These can provide a more immediate massive parallelism, implemented directly by the underlying physical properties of the substrate being exploited. There is a wide range of problem-specific analogue computers, but there are also general purpose analogue computers. Mills has built implementations of Ruebel's extended analog computer [Rubel 1993], where the computational substrate is a conductive sheet [Mills et al 2006] [Mills 2008b] and has developed a computational metaphor to aid its programming by analogy [Mills 2008a]. It is clear, however, that there is still challenging work to be done to provide the level of support for programming such devices as we currently enjoy in the classical realm. What are suitable models for programming by analogy? How expressive can these models be?

3.5 Embodiment and *in materio* computing

We choose to distinguish *embodiment* (real-time close coupling of a computational device with its complex environment) from *in materio* computing (computation directly by physical and chemical processes of a complex substrate, with little or no abstraction to a virtual machine). Most authors choose to conflate these properties (for example [MacLennan 2008]), requiring embodiment to be physical. However, separation of these concepts allows us to consider *virtual* embodied systems [Stepney 2007], for example, the embodiment of software agents closely coupled with a complex but virtual internet environment, and to consider non-embodied *in materio* computation (computation by a physical medium [Miller & Downing 2002] [Harding et al 2008]), where the computation performed by that medium need not be real-time, nor closely coupled with a complex environment.

With this view, embodied computation can be implemented either as *embedded* computation (a classical computer embedded in a body that interacts with the environment via classical sensors and actuators), or as *in materio* computation (computation by the physical properties and processes of its body). Biological systems, “wetware”, are embodied (where their complex environment comprises the physical world, other biological organisms, and other parts of the organism that the biological system of concern, such as a cell, inhabits) and *in materio* (implemented in a complex material substrate that directly performs the relevant computation and information processing). Hybrid embodiment is becoming more common, where some of the processing is embedded (a classical embedded computational “brain”), and some is *in materio* (any computation performed directly by the rest of the physical body). Mills’ implementation of the extended analog computer [Mills et al 2006] [Mills 2008a] [Mills 2008b] has this hybrid nature too: a combination of explicit logic circuits and implicit analogue computation. In the hybrid embodied case, the behaviours of the sensors and actuators are not so easily separated from the computation: they too are part of the body and can be considered to be performing *in materio* computation (for example, complex signal transduction, or even the case of a “slinky” toy spring that can “walk” downstairs with no controller or motor). An excellent review of some of these aspects applied to robotics can be found in [Pfeifer et al 2007]. The fact that controller, body, sensors and actuators cannot easily be separated (and co-evolve in biological systems) is being investigated in information theoretic terms by Polani and colleagues, for example [Olsson et al 2006] [Polani et al 2007].

The direct computation performed *in materio* can be extremely efficient compared to simulating similar processes classically: the material merely does what comes naturally, on timescales directly controlled by the physical laws. [MacLennan 2008] discusses how physical systems can efficiently implement massively parallel diffusion, sigmoid functions, negative feedback, and randomisation. Another example is the so called “protein folding problem”, intractable classically, but computable by physical proteins in a few seconds. The challenge here is to discover how to exploit various complex material substrates to do computation [Bechmann et al 2007] [Stepney 2008], to explore the range of possibilities exhibited, and to discover what (if any) generalisations can be made. For this efficiency comes with a price: we currently have to trade off efficiency for *abstraction* (the computation is performed directly by the physical medium, with little or no abstraction into virtual levels, which can be argued to be essential for certain forms of computation [Sloman & Chrisley 2003]), and (often) for *universality*. So the broader challenge is to discover what abstractions *can* be made: what are appropriate abstract models of classes of *in materio* computation? In other words, what kind of *in materio* computations have at least some degree of substrate-independence? (They might require some complex non-linear dynamical behaviour, but that kind of behaviour might be exhibited by many substrates; [MacLennan 2004] calls this *multiple instantiability*). How many essentially different kinds of models are there? Do any provide universality? Do any provide the self-referentiality that allows meta-programming and Strange Loops [Hofstadter 2007]? How expressive can these models be? Is *material* embodiment essential to ground the semantics of the system [Rosen 1991], or can virtual systems exhibit the same kind of properties?

3.6 Growth and self assembly

Biological systems are able to *self assemble* and *grow*: their environment provides a flux of energy, matter, and information, which they can process to form further material computational substrate (for example, more cells).

Ideas from biological growth and development are beginning to influence evolutionary computation. The fact that biological organisms have a complex non-linear mapping from their genotype (DNA) to their phenotype (body) has been taken on board in “evo-devo” algorithms: evolutionary algorithms that “evolve” a digital genotype, that then develops, or “grows” into the desired phenotype. Here the genotype is a “generative encoding”: it encodes an *indirect* description of the result, which when decoded *generates* the phenotype. Genetic Programming [Koza 1992], where the genotype is a computer program, and the phenotype is the result of the execution of that program, may be thought of as an early instance of this idea. Newer ideas focus on kinds of generative rules for how the genotype decodes to the phenotype. Cartesian Genetic Programming [Miller & Thomsom 2000] decodes a string of numbers (genotype) into a network of components by interpreting the numbers as descriptions of the network connections, and as indexes into a catalogue of components. Grammatical Evolution [O'Neill & Ryan 2003] decodes a string of numbers into a computer program by interpreting the numbers as indexes into a BNF grammar of the programming language. L-Systems [Prusinkiewicz & Lindenmayer 1990], a parallel generative grammar originally used to describe plant growth, are a popular form of encoding for a range of phenotypes [Hornby & Pollack 2001a] [Hornby & Pollack 2001b]. In all these cases, the genotype is subject to evolution (mutation, crossover, etc), whilst the generated phenotype is evaluated for fitness.

The complex non-linear genotype-phenotype mapping can be considered as an attractive feature to add to evolutionary algorithms, or the generative process can be considered as the main focus, in order to “grow” software and other computational structures [Kumar & Bentley 2003] [Miller et al 2005] [Rieffel & Pollack 2005]. Recently, Christopher Alexander, the architect whose work inspired Software Engineering Pattern Languages [Alexander et al 1977] [Gamma et al 1995], has more recently been working on morphological ideas in architecture, and looking for a computational description of them [Alexander 2009].

Computational growth and development ideas are not restricted to software. Much exciting work is taking place in the area of growing and self-assembling nanotech computational artefacts. Adelman’s original DNA computer idea [Adleman 1994] used DNA synthesis to grow the computer, and the chemistry of DNA matching to perform the computation (path finding). Sometimes the growth process *is* the computation (growing, or self-assembling, certain structures or patterns, etc); more exotic approaches grow a system that then computes ("programmable matter"). There is a vast literature; see, for example, [Fujibayashi et al 2007] [Goldstein & Mowry 2004] [Jonoska 2006] [MacLennan 2008] [Rothemund 2006] [Seeman 2007]. As with all these non-classical approaches, there are no hard-and-fast dividing lines: it is not (and need not be) clear where growth stops and computation begins.

The challenges here are numerous, and only beginning to be articulated. How can software systems be made rich enough so that new structures and behaviours, and new *kinds* of structures and behaviours, can "grow" naturally out of these systems? Dynamical systems theory is a staple of non-linear systems analysis, yet it assumes that dimensionality is given, and constant, and that the dimensions are (relatively) homogeneous. What are the mathematical techniques for modelling systems whose dimensionality changes and grows in essentially unpredictable (because contingent and emergent) ways as the system develops, where the dimensions are hybrid (a mix of discrete and continuous), and where new dimensions may be of new types?

3.7 Other GC7 activities

The reviews here by no means cover all aspects of GC7. In particular, we have concentrated on what might be called "hard" non-classical computing, perhaps somewhat betraying the main interests of the authors. Additionally, there are many "softer" non-classical issues. For example, the "Thinking Through Computing" workshop (§2) is part of a process for bridging the gap (nay, gulf?) between the Computer Science and the Arts and Humanities view of computing, to promote a broader view of computing. As computers become more embedded and ubiquitous, being used in ways we cannot yet even conceive, views of what computers are, what they are for, and how they should and could be used, will inevitably become ever broader.

Criteria for success of an open-ended challenge like GC7 are difficult to formulate. Nevertheless, one activity of the GC7 community should be to develop a set of partial success criteria, some directed to particular journeys (covering development of theory, and development of implementations), some directed to the integration and unification of different journeys. These criteria should allow a good balance between providing discipline and structure to the challenge, and giving free rein to novel ideas.

4 IMPLEMENTATIONS OF REACTION-DIFFUSION COMPUTERS

Many discussions of non-classical computation can remain highly theoretical and conceptual. After all, exotic space-time singularities are not easily come by to implement general relativistic computations. Even quantum computers are proving challenging to implement. We have mentioned above some material realisations: Mills' implementation of Rubel's extended analog computer, and various embodied DNA devices. Here we review one class of implementations in some detail: reaction-diffusion computers. The class of reaction-diffusion computers can be implemented in a wide range of substrates, from single-electron nano circuits, through chemicals, to slime moulds. The larger, and slower, devices provide a relatively cheap way to investigate the general principles of this computational paradigm.

4.1 Reaction-diffusion chemical computers

A reaction-diffusion computer [Adamatzky 2001] [Adamatzky et al 2005] is a spatially extended chemical system, which processes information using interacting growing patterns of excitable and diffusive waves. In reaction-diffusion processors, both the data and the results of the computation are encoded as concentration profiles of the reagents. The computation is performed via the spreading and interaction of wave fronts.

The reaction-diffusion computers are parallel because myriads of their micro-volumes update their states simultaneously, and molecules diffuse and react in parallel. Liquid-phase chemical media are wet-analogues of massively-parallel (millions of elementary processors in a small chemical reactor) and locally-connected (every micro-volume of the medium changes its state depending on states of its closest neighbours) processors. They have parallel input and outputs, e.g. optical input -- control of initial excitation dynamics by illumination masks, and output is parallel because the concentration profile representing results of the computation is visualised by indicators. The reaction-diffusion computers are fault-tolerant and capable of automatic reconfiguration, namely if we remove some quantity of the computing substrate, the topology is restored almost immediately.

Reaction-diffusion computers are based on three principles of physics-inspired computing [Margolus 1984]. First, physical action measures the amount of information: we exploit active processes in non-linear systems and interpret the dynamics of the systems as computation. Second, physical information travels only a finite distance: this means that computation is local and we can assume that the non-linear medium is a spatial arrangement of elementary processing units connected locally, i.e. each unit interacts with its closest neighbours. Third, nature is governed by waves and spreading patterns: computation is therefore spatial.

Cellular automata and numerical models of reaction-diffusion computers are well-described in [Adamatzky, 2001] [Adamatzky et al., 2005]; real-world implementations of reaction-diffusion computer are worth further mention.

Image processing: The first ever experimental prototype of a chemical image processor was designed by Kuhnert, who used light-sensitive modification of the Belousov-Zhabotinsky reaction to implement basic operations of image processing [Kuhnert 1986] [Kuhnert et al 1989]. Experimental reaction-diffusion processors were implemented capable of contouring, negative and positive image alteration, removal or enhancement of minor features, image filtration and image restoration; see [Adamatzky 2001] [Adamatzky et al 2005] and references therein.

Graph problems/path planning: Chemical processor for approximation of a shortest path in planar labyrinths or rooms with obstacles were produced by Steinbock et al. [1995], Agladze et al. [1997], and Rambidi & Yakovenchuk [2001]. In all versions of the processors impassable barriers and obstacles are represented either by physically removed parts of the substrate, drops of chemical inhibitors, or inhibiting illumination. Steinbock & Agladze's implementations employed classical trigger waves, initiated firs at the source and then at the destination, two scenarios of wave propagation were recorded on camera, and analysed. A shortest path was extracted from topologies of intersecting wave-fronts. Two passes of wave propagation are unavoidable, as was demonstrated in theoretical studies [Adamatzky 1996]. Rambidi & Yakovenchuk have used phase waves, where propagation was controlled by lights; additional image processing was required nevertheless. The problem of shortest path calculation was solved in laboratory prototypes of excitable medium processor: Belousov-Zhabotinsky medium interfaced with excitable cellular automaton [Adamatzky & De Lacy Costello 2002], and precipitating chemical medium, palladium processor, [Adamatzky et al 2003].

Computational geometry: The Voronoi diagram, or a tessellation of a plane from a given finite planar set, is a classical hard problem of computational geometry. The planar data set is represented by a configuration of drops of one reactant, and another reactant is contained within the gel substrate. The data-reactant diffuses and forms a coloured precipitate when reacting with the substrate-reactant. When waves of data-reactants meet up they exhaust the substrate-reactant and no precipitate is formed. The uncoloured sites of the medium represent edges of the Voronoi diagram; see descriptions of experimental prototypes in [Tolmachev & Adamatzky, 1996] [De Lacy Costello et al 2004] [Adamatzky et al 2005]. In contrast to conventional approaches, even those implemented in massively-parallel VLSI processors, the exact type of data-objects does not affect the complexity of Voronoi diagram computation in reaction-diffusion processors. The same precipitating chemical medium processes planar sets, sets of arbitrary planar objects [Adamatzky & De Lacy Costello, 2003], and even calculates the skeleton [Adamatzky & Tolmachev 1997] [Adamatzky et al 2002] of a planar shape in linear time.

Universal computation: There two classes of universal reaction diffusion computers (computers experimentally implemented in spatially extended chemical media that implement a functionally complete set of logical functions): geometrically constrained and free-space devices.

In geometrically-constrained devices, excitation waves travel along channels, interact with each other at the junctions of the channels, and execute logical operations as the result of their interactions; see e.g. [Toth & Showalter 1995] [Sielewiesiuk & Gorecki 2001] [Adamatzky & De Lacy Costello 2002a] [Motoike & Adamatzky 2004]. Constrained by stationary wires and gates, chemical processors mimic conventional silicon computing devices, their novelty is only in the chemical medium implementation.

Free-space[11], or architectureless, chemical computers employ the paradigm of collision-based computing; see the overview in [Adamatzky 2003]. These originated from the computational universality of Game of Life [Berlekamp et al 1982], conservative logic, and the billiard-ball model [Fredkin & Toffoli 1982] [Margolus 1984]. A collision-based, or dynamical, computation employs mobile compact finite patterns, mobile self-localised excitations, or simply localisations, in an active non-linear medium. Information values (for example, truth values of logical variables) are given either by absence (or presence) of the localisations, or by other parameters of the localisations. The localisations travel in space, and do computation when they collide with each other. There are no predetermined stationary wires, a trajectory of the travelling pattern is a momentary wire. Almost any part of the medium space can be used as such a wire. Localisations can collide anywhere within a space: there are no fixed positions at which specific operations occur, nor location-specified gates with fixed operations. The localisations undergo transformations, form bound states, and annihilate or fuse when they interact with other mobile patterns. Information values of localisations are transformed as a result of collision, and thus a computation is implemented [Adamatzky 2003].

Chemical laboratory implementation of collision-based computers is possible thanks to a photosensitive sub-excitable Belousov-Zhabotinsky medium that exhibits propagating wave-fragments that preserve their shapes over substantial periods of time. The presence (or absence) of wave-fragments is interpreted as a truth value of a Boolean variable. When two or more wave fragments collide they may annihilate, fuse, split or deviate from their original paths; the corresponding values of the logical variables are changed, and logical gates are realised as a result of the collision. The exact types of logical gates implemented in the collisions between excitation wave-fragments depend on the size of the wave-fragments, their phase, and the relative configuration of their velocity vectors at the moment of collision [Adamatzky 2004] [De Lacy Costello & Adamatzky 2005]. Complicated cascaded logical circuits can be designed in hybrid systems, where excitable chemical media co-evolve with massively-parallel automata networks [Bull et al 2007] [Toth et al 2008]. There is a high chance that soon we will be able to produce laboratory prototypes of Belousov-Zhabotinsky arithmetical processors, as demonstrated already in scoping computational experiments with binary adders in excitable media [Zhang & Adamatzky 2008].

[11] Term invented by Prof. Jonathan Mills

Robotics: Spatially extended excitable chemical media can accept parallel optical inputs, perform computation simultaneously in many sites of their substrates, and, thanks to indicators, can generate outputs in parallel. This makes them unique candidates for the role of decentralised 'wetware' controllers for robotic devices. We have demonstrated the concept in two experimental prototypes of chemical media coupled with hardware robotic devices. To control navigation of a wheeled robot, we installed a chemical reactor on board the robot, allowing the robot to observe the space-time dynamics of the excitation waves. When the waves are triggered by direct application of chemicals, or a light, the robot detects the relative position of the stimulator from the topology of the excitation waves, updates parameters of its movement, and thus progresses towards the source of stimulation [Adamatzky et al 2004]. To investigate closed-loop interaction between the chemical medium and a robot, we interfaced a Belousov-Zhabotinsky medium with a robotic hand in such a manner that excitation wave fronts propagating in the medium triggered movement of the hand's fingers, which in turn physico-chemically perturbed the Belousov-Zhabotinsky medium [Yokoi et al 2004]. Complex patterns of non-trivial hand finger motions were demonstrated experimentally. Large-scale simultaneous manipulation of several objects by an excitable chemical medium can be achieved if every micro-volume of the medium is coupled with a micro-actuator; propagating wave-fronts generate force fields, which can sort, orient, transport and assemble arbitrarily-shaped objects [Adamatzky et al 2005a].

4.2 Encapsulated reaction-diffusion computers: Physarum machines

There still remains a range of problems where chemical reaction-diffusion processors cannot cope without the external support from conventional silicon-based computing devices. The shortest path computation is one such problem. As discussed above, one can use excitable media to outline a set of all collision-free paths in a space with obstacles, but to select and visualise the shortest path amongst all these possible paths, one needs to use external computing power. Experimental setups that claim to directly compute a shortest path in a chemical medium [Steibock et al 1995] [Agladze et al 1997] employ external computing resources to store time-lapsed snapshots of propagating wave-fronts and to analyse the dynamics of the wave-front propagation. Such usage of external resources dramatically reduces the fundamental value of computing with propagating patterns. Graph-theoretical computations pose even more difficulties for spatially-extended non-linear computers. For example, one can compute the Voronoi diagram of a planar set, but cannot invert this diagram [Adamatzky & De Lacy Costello 2005].

Let us consider a spanning tree, the most famous of classical proximity graphs. Given a set of planar points one wants to connect the points with edges, such that the resultant graph has no cycles, and there is a path between any two points of the set. So far, no algorithms of spanning tree construction have been experimentally implemented in spatially extended non-linear chemical systems. This is caused mainly by the uniformity of spreading wave-fronts, their inability to sharply select directions toward locations of data points, and also because excitable systems usually do not form stationary structures. To overcome these difficulties, we should allow reaction-diffusion computers to be geometrically self-constrained while still capable operating in geometrically unconstrained, architectureless, or 'free', space [Adamatzky 2007]. We have demonstrated that the vegetative state, or plasmodium, of *Physarum polycephalum* is a reaction-diffusion system constrained by a membrane, capable of solving graph-theoretical problems [Adamatzky 2007] [Adamatzky 2008]. When raised on a nutrient-poor substrate, the plasmodium of *Physarum polycephalum* can also implement basic operations of the Kolmogorov-Uspensky machine, mother of modern storage modification machines, thus being capable of general purpose computation [Adamatzky 2008a].

5. CONCLUSION: BEWARE "REGRESSION TO THE CLASSICAL"

As we can see from this review, research into a wide range of aspects of non-classical computation is prevalent, and is gaining maturity. However, we end on a note of caution. We have noticed certain distressing tendencies of a subset of non-classical computation researchers, which we dub "regression to the classical" (this section is deliberately light on references, to protect the accused, who include ourselves at times).

One particular such tendency is to reimplement a universal Turing machine in whatever weird substrate is being considered as a computational medium. We have argued [Stepney 2008] that this tells us very little new about computation, and even less about non-classical computation. As noted above (§4.1), the only novelty of this is in the choice of medium. Non-classical research into strange substrates should uncover what computational behaviour they can do naturally, not merely torture them to do what silicon can do better.

Another tendency is to develop non-classically inspired theoretical models that are essentially indistinguishable from classical models. For example, DNA or enzymes (proteins) routinely get modelled as strings, and the operations that the molecules perform then get modelled as string operators. Then the string model is exhaustively investigated from a theoretical standpoint, often to demonstrate that it is Turing universal. But what often happens along the way is that the abstractions and approximations to build that model go a step too far, and lose any connection with their original inspiration (for example, enzyme binding being modelled as string operations with completely unbiological characteristics).

This tendency is explicable. After a hard day hewing through the non-classical thickets, with no map and precious little sense of direction, it is a relief to retreat to one's comfort zone of well-understood classical results. But this is not what GC7 is about. GC7 is about drawing that map, charting the currently undiscovered territories, banishing the dragons, and thereby establishing new comfort zones.

REFERENCES.

[Abramsky 2005] S. Abramsky. What are the fundamental structures of concurrency? We still don't know! In *Algebraic process calculi: the first 25 years and beyond*, BRICS Notes Series NS-05-03, June 2005

[Abramsky 2006] S. Abramsky. Socially Responsive, Environmentally Friendly Logic, Truth and Games. In Aho, Pietarinen, eds., *Essays in Honour of Gabriel Sandu*, *Acta Philosophica Fennica*, Helsinki, 2006

[Adamatzky 1996] A Adamatzky. Computation of shortest path in cellular automata. *Math. Comput. Modelling* **23**: 105–113 1996.

[Adamatzky & Tolmachev 1997] A Adamatzky, D Tolmachiev. Chemical processor for computation of skeleton of planar shape. *Adv. Mater. Opt. Electron*. **7**:135–139 1997.

[Adamatzky 2001] A Adamatzky. *Computing in Nonlinear Media and Automata Collectives*. IoP Publishing, 2001

[Adamatzky & De Lacy Costello 2002] A Adamatzky, BPJ De Lacy Costello. Collision-free path planning in the Belousov-Zhabotinsky medium assisted by a cellular automaton. *Naturwissenschaften* **89**:474–478 2002

[Adamatzky & De Lacy Costello 2002a] A Adamatzky, BPJ De Lacy Costello. Experimental logical gates in a reaction-diffusion medium: the XOR gate and beyond. *Phys. Rev. E* **66**:046112 2002

[Adamatzky et al 2002] A Adamatzky, B De Lacy Costello, NM Ratcliffe. Experimental reaction-diffusion pre-processor for shape recognition. *Phys. Lett. A* **297**:344–352 2002

[Adamatzky 2003] A Adamatzky. (ed.) *Collision Based Computing.* Springer, 2003.

[Adamatzky et al 2003] A Adamatzky, B De Lacy Costello, C Melhuish, N Ratcliffe. Experimental reaction-diffusion chemical processors for robot path planning. *J. Intell. Robot. Syst*. **37**:233–249 2003

[Adamatzky & De Lacy Costello 2003] A Adamatzky, BPJ De Lacy Costello. On some limitations of reaction-diffusion computers in relation to a Voronoi diagram and its inversion. *Phys. Lett. A* **309**:397–406 2003

[Adamatzky 2004] A Adamatzky. Collision-based computing in Belousov-Zhabotinsky medium. *Chaos Solitons Fractals* **21**:1259–1264 2004

[Adamatzky et al 2004] A Adamatzky, B De Lacy Costello, C Melhuish, N Ratcliffe. Experimental implementation of mobile robot taxis with onboard Belousov-Zhabotinsky chemical medium. *Mater. Sci. Eng. C* **24**:54–548 2004

[Adamatzky et al 2005] A Adamatzky, B De Lacy Costello, T Asai. *Reaction-Diffusion Computers*. Elsevier, 2005

[Adamatzky et al 2005a] A Adamatzky, B De Lacy Costello, S Skachek, C Melhuish. Manipulating objects with chemical waves: Open loop case of experimental Belousiv-Zhabotinsky medium. *Phys. Lett. A* 2005

[Adamatzky & Teuscher 2006] A Adamatzky, C Teuscher, eds. *From Utopian to Genuine Unconventional Computers*. Luniver Press 2006

[Adamatzky 2007] A Adamatzky. Physarum machines: encapsulating reaction-diffusion to compute spanning tree. *Naturwisseschaften* 2007

[Adamatzky et al 2007] A Adamatzky, L. Bull, B. De Lacy Costello, S Stepney, C Teuscher, eds. *Unconventional Computing 2007, Bristol, UK, July 2007*. Luniver Press 2007

[Adamatzky 2008] A Adamatzky. Physarum machine: implementation of a Kolmogorov-Uspensky machine on a biological substrate. *Parallel Processing Letters* 2008

[Adamatzky 2008a] A Adamatzky. Approximation of spanning trees in plasmodium of Physarum polycephalum, *Kybernetes* 2008

[Adamatzky et al 2008] A Adamatzky, R Alonso-Sanz, A Lawniczak, GJ Martinez, K Morita, T Worsch, eds. *Automata 2008: Theory and Applications of Cellular Automata*. Luniver Press, 2008

[Adleman 1994] LM Adleman. Molecular Computation of Solutions To Combinatorial Problem. *Science* **266**:1021–1024 1994

[Agladze et al 1997] K Agladze, N Magome, R Aliev, T Yamaguchi, K Yoshikawa. Finding the optimal path with the aid of chemical wave. *Physica D* **106**:247–254 1997

[Alexander et al 1977] C Alexander, S Ishikawa, M Silverstein, M Jacobson, I Fiksdahl-King, S Angel. *A Pattern Language: towns, buildings, construction*. Oxford University Press, 1977

[Alexander 2009] C Alexander. Harmony-Seeking Computations. *Int. J. Unconventional Computing* 2009 (in press)

[Andrews et al 2008] PS Andrews, AT Sampson, JM Bjørndalen, S Stepney, J Timmis, DN Warren, PH Welch. Investigating Patterns for the Process-Oriented Modelling and Simulation of Space in Complex Systems. *ALife XI, Winchester, UK, August 2008*. MIT Press 2008

[Banzhaf et al 2006] W Banzhaf, G Beslon, S Christensen, JA Foster, F Képès, V Lefort, JF Miller, M Radman, JJ Ramsden. From Artificial Evolution to Computational Evolution: a research agenda. *Nature Reviews Genetics* **7**:729–735 2006

[Bechmann et al 2007] M Bechmann, JA Clark, A Sebald, S Stepney. Unentangling nuclear magnetic resonance computing. *Unconventional Computing 2007, Bristol, UK, July 2007*, pp1–18. Luniver Press, 2007

[Berlekamp et al, 1982] ER Berlekamp, JH Conway, RL Guy. *Winning Ways for your Mathematical Plays, Vol. 2*. Academic Press, 1982

[Blakey 2007] E Blakey. On the computational complexity of physical computing systems. *Unconventional Computing 2007, Bristol UK*, pages 199–220. Luniver Press, 2007

[Bull et al., 2007] L Bull, A Budd, C Stone, I Uroukov, B De Lacy Costello, A Adamatzky. Towards unconventional computing through simulated evolution: Learning Classifier System control of non-linear media. *Artificial Life* 2007, in press

[Calude et al 2006] CS Calude, MJ Dinneen, G Paun, G Rozenberg, S Stepney, eds. *Unconventional Computation, UC 2006, York, UK, September 2006*. LNCS **4135**. Springer, 2006.

[Copeland 2002] BJ Copeland. Accelerating Turing Machines. *Minds Mach*. **12**(2):281-300 May 2002

[De Lacy Costello et al 2004] B De Lacy Costello, A Adamatzky, N Ratcliffe, AL Zanin, AW Liehr, HG Purwins. The formation of Voronoi diagrams in chemical and physical systems: experimental findings and theoretical models. *Int. J. Bifurcat. Chaos* **14**:2187–2210 2004

[De Lacy Costello & Adamatzky 2005] B De Lacy Costello, A Adamatzky. Experimental implementation of collision-based gates in Belousov–Zhabotinsky medium. *Chaos, Solitons & Fractals* **25**:535–544 2005

[Fredkin & Toffoli 1982] E Fredkin, T Toffoli. Conservative logic. *Int. J. Theor. Phys*. **21**:219–253 1982

[Fujibayashi et al 2007] K Fujibayashi, R Hariadi, SH Park, E Winfree, S Murata. Toward Reliable Algorithmic Self-Assembly of DNA Tiles: A Fixed-Width Cellular Automaton Pattern. *Nano Lett*. 10.1021/nl0722830 2007

[Gamma et al 1995] [2] E Gamma, R Helm, RE Johnson, J Vlissides. *Design Patterns: elements of reusable object-oriented software*. Addison Wesley, 1995

[Goldstein & Mowry 2004] SC Goldstein, TC Mowry. Claytronics: A scalable basis for future robots. *RoboSphere 2004*, November, 2004.

[Harding et al 2008] SL Harding, JF Miller, EA Rietman. Evolution in Materio: Exploiting the Physics of Materials for Computation. *Int. J. Unconventional Computing* **4**(2):155-194 2008

[Hoare 1985] CAR Hoare. *Communicating Sequential Processes*. Prentice Hall, 1985

[Hofstadter 2007] DR Hofstadter. *I am a Strange Loop*. Basic Books, 2007

[Hogarth 1992] M Hogarth. Does General Relativity allow an observer to view an eternity in a finite time? *Foundations of Physics Letters* **5**:73–81:1992

[Hornby & Pollack 2001a] GS Hornby, JB Pollack. Body-Brain Co-evolution Using L-systems as a Generative Encoding. GECCO 2001, pp868–875. Morgan Kaufmann 2001

[Hornby & Pollack 2001b] GS Hornby, JB Pollack. The Advantages of Generative Grammatical Encodings for Physical Design. CEC 2001, pp600–607. IEEE Press 2001

[IJUC 2006] *Int. J. Unconventional Computing* **2**(4) 2006 Special issue 1 on GC7 workshop

[IJUC 2007] *Int. J. Unconventional Computing* **3**(3) 2007 Special issue 2 on GC7 workshop

[IJUC 2008a] *Int. J. Unconventional Computing* **4**(1) 2008 Special issue on UC'06 workshop

[IJUC 2008b] *Int. J. Unconventional Computing* **4**(3) 2008 Special issue on UC'06 conference

[IJUC 2009] *Int. J. Unconventional Computing* **5** 2009 Special issue on hypercomputation (in press)

[JCA 2009] *J. Cellular Automata* 2009 Special issue on Automata 2008 conference (in prep)

[Jonoska 2006] N Jonoska. Theoretical and Experimental DNA Computation. *Genetic Programming and Evolvable Machines* **7**(3): 287–291 2006

[Koza 1992] JR Koza. *Genetic Programming: on the programming of computers by means of natural selection*. MIT Press, 1992

[Kuhnert 1996] L Kuhnert. Photochemische Manipulation von chemischen Wellen. *Naturwissenschaften* **76**:96–99 1986

[Kuhnert et al 1989] L Kuhnert, K Agladze, V Krinsky. Image processing using light-sensitive chemical waves. *Nature* **337**:244–247 1989

[Kumar & Bentley 2003] S Kumar, PJ Bentley, eds. *On Growth, Form and Computers*. Elsevier, 2003

[Landauer 1961] R Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development* **5**(3):183–191 1961

[Lloyd 2000] S Lloyd. Ultimate physical limits to computation. *Nature* **406**:1047–1054 2000

[MacLennan 2004] BJ MacLennan. Natural computation and non-Turing models of computation. *Theoretical Computer Science* **31**(1–3):115–145 June 2004

[MacLennan 2008] BJ MacLennan. Aspects of Embodied Computing. Technical Report UT-CS-08-610, University of Tennessee, March 2008

[Margolus 1984] N Margolus. Physics-like models of computation. *Physica D* **10**:81–95 1984

[Miller & Thomson 2000] JF Miller, P Thomson. Cartesian Genetic Programming. *EuroGP 2000*, LNCS **1802**:121–132. Springer, 2000

[Miller & Downing 2002] JF Miller, K Downing. Evolution in materio: Looking beyond the silicon box. *Proc. NASA/DoD Conf. on Evolvable Hardware, Alexandria, VA, USA, July 2002*, pp167–176. IEEE Press, 2002

[Miller et al 2005] JF Miller, G Hornby, S Kumar, eds. Scalable, Evolvable, Emergent Design and Developmental Systems (SEEDS) Workshop, *GECCO 2005, Washington DC* 2005

[Mills 2008a] JW Mills. The nature of the extended analog computer. *Physica D* 2008

[Mills 2008b] JW Mills. The architecture of an extended analog computer core. *UCAS-4, Austin, TX, USA* 2008

[Mills et al 2006] JW Mills, B Himebaugh, et. al., "Empty Space" Computes: The Evolution of an Unconventional Supercomputer. *Proc. ACM Computing Frontiers Conf*. 2006

[Milner 1999] R Milner. *Communicating and Mobile Systems: the π-Calculus*. Cambridge University Press, 1999

[Motoike & Adamatzky 2004] IN Motoike, A Adamatzky. Three-valued logic gates in reaction-diffusion excitable media. *Chaos Solitons Fractals* **24**:107–114 2004

[NCJ 2008] *Natural Computing* **7**(1) 2008 Special issue on UC'06 conference

[O'Neill & Ryan 2003] M O'Neill, C Ryan. *Grammatical Evolution: evolutionary automatic programming in an arbitrary language*. Kluwer, 2003

[Olsson et al 2006] L Olsson, CL Nehaniv, D Polani. From Unknown Sensors and Actuators to Actions Grounded in Sensorimotor Perceptions. *Connection Science*, **18**(2):121–144 2006

[Pfeifer et al 2007] R Pfeifer, M Lungarella, F Iida. Self-organization, embodiment, and biologically inspired robots. Science **318**:1088–1093 November 2007

[PhysD 2008] Physica D **237** 2008 Special issue on UC Santa Fe workshop

[Polack et al 2005] F Polack, S Stepney, H Turner, P Welch, F Barnes. An Architecture for Modelling Emergence in CA-Like Systems. *ECAL 2005*, LNAI **3630**:427-436. Springer 2005

[Polack et al 2008] F Polack, S Stepney, AT Sampson, J Timmis, T Hoverd. Complex Systems Models: Engineering Simulations. *ALife XI, Winchester, UK, August 2008*. MIT Press 2008

[Polani et al 2007] D Polani, O Sporns, M Lungarella. How Information and Embodiment Shape Intelligent Information Processing. *50 Years of Artificial Intelligence*. LNCS **4850**:99–111. Springer 2007

[Prusinkiewicz & Lindenmayer 1990] P Prusinkiewicz, A Lindenmayer. *The Algorithmic Beauty of Plants*. Springer, 1990

[Rieffel & Pollack 2005] J Rieffel, J Pollack. Evolutionary Fabrication: the emergence of novel assembly methods in artificial ontogenies. *GECCO 2005*, pp 265–272. ACM 2005

[Rosen 1991] R Rosen. *Life Itself: a comprehensive enquiry into the nature, origin, and fabrication of life*. Columbia University Press, 1991

[Rothemund 2006] PWK Rothemund. Folding DNA to create nanoscale shapes and patterns. *Nature* **440**:297–302 2006

[Rubel 1993] LA Rubel. The Extended Analog Computer. *Advances in Applied Mathematics* **14**:39–50, 1993

[Seeman 2007] NC Seeman. An Overview of Structural DNA Nanotechnology. *Mol. Biotech*. **37**:246–257 2007

[Sielewiesiuk & Gorecki 2001] J Sielewiesiuk, J Gorecki. Logical functions of a cross-junction of excitable chemical media. *J. Phys. Chem. A* **105**:189–8195 2001

[Sloman & Chrisley 2003] A Sloman, R Chrisley. Virtual Machines and Consciousness. *Journal of Consciousness Studies* **10**(4-5):113–172 2003

[Steinbock et al 1995] O Steinbock, A Toth, K Showalter. Navigating complex labyrinths: optimal paths from chemical waves. *Science* **267**:868–871 1995.

[Stepney et al 2005a] S Stepney, SL Braunstein, JA Clark, A Tyrrell, A Adamatzky, RE 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

[Stepney et al 2005b] S Stepney, RE Smith, J Timmis, AM Tyrrell, MJ Neal, ANW Hone. Conceptual Frameworks for Artificial Immune Systems. *Int. J. Unconventional Computing*. **1**(3):315–338, July 2005

[Stepney et al 2006a] S Stepney, SL Braunstein, JA Clark, A Tyrrell, A Adamatzky, RE Smith, T Addis, C Johnson, J Timmis, P Welch, R Milner, D Partridge. Journeys in Non-Classical Computation II: Initial journeys and waypoints. *Int. J. Parallel, Emergent and Distributed Systems*. **21**(2):97–125, April 2006

[Stepney et al 2006b] S Stepney, F Polack, H Turner. Engineering Emergence. *ICECCS'06, Stanford, CA, USA, August 2006*. pages 89–97. IEEE Press 2006

[Stepney 2007] S Stepney. Embodiment. In Flower, Timmis, eds. *In Silico Immunology*, chapter 12, Springer, 2007

[Stepney et al 2007] S Stepney, T Clarke, P Young. PLAZZMID: an agent-based architecture inspired by bee and bacterial genomes. *ECAL 2007, Lisbon, Portugal, September 2007*. LNAI **4648**:1151–1160 Springer 2007

[Stepney 2008] S Stepney. The Neglected Pillar of Material Computation. *Physica D* **237**, July 2008

[Stepney 2009] S Stepney. Non-Classical Hypercomputation. *Int. J. Unconventional Computing* **5** 2009 (in press)

[Tempesti & Teuscher 2003] G Tempesti, C Teuscher. Biology Goes Digital: An array of 5,700 Spartan FPGAs brings the BioWall to 'life'. XCell Journal **47**:40–45, Fall 2003

[Timmis et al 2006] J Timmis, M Amos, W Banzhaf, A Tyrrell. “Going back to our Roots": Second Generation Biocomputing. *Int. J. Unconventional Computing*. **2**(4):349–382 2006

[Timmis et al 2008] J Timmis, E Hart, A Hone, M Neal, A Robins, S Stepney, A Tyrrell. Immuno-Engineering. *2nd IFIP International Conference on Biologically Inspired Collaborative Computing, Milan, Italy, September 2008*. IEEE Press, 2008

[Tolmachev & Adamatzky 1996] D Tolmachiev, A Adamatzky. Chemical processor for computation of Voronoi diagram. *Adv. Mater. Opt. Electron*. **6**:191–196 1996

[Toth & Showalter 1995] A Toth, K Showalter. Logic gates in excitable media. *J. Chem. Phys*. **103**:2058–2066 1995

[Toth et al 2008] R Toth, C Stone, B De Lacy Costello, A Adamatzky, L Bull. Dynamic Control and Information Processing in the Belousov-Zhabotinsky Reaction using a Co-evolutionary Algorithm. *Journal of Chemical Physics* 2008 (in press)

[Rambidi & Yakovenchuk 2001] NG Rambidi, D Yakovenchuk. Chemical reaction-diffusion implementation of finding the shortest paths in a labyrinth. *Phys. Rev. E* **63**:026607 2001

[Wegner 1997] P Wegner. Why interaction is more powerful than algorithms. *Comm. ACM* **40**(5):81–91 1997

[Welch & Barnes 2005] PH Welch, FRM Barnes. Communicating mobile processes: introducing occam-π. *25 Years of CSP*. LNCS **3525**:175–210. Springer 2005

[Welch et al 2006] PH Welch, FRM Barnes, FAC. Polack. Communicating complex systems. *ICECCS'06, Stanford, CA, USA, August 2006*. pages 107–120. IEEE Press 2006

[Wheeler 1990] JA Wheeler. Information, physics, quantum: The search for links. In WH Zurek, ed, *Complexity, Entropy and the Physics of Information*. Addison Wesley, 1990

[Yokoi et al 2004] H Yokoi, A Adamatzky, B De Lacy Costello, C Melhuish. Excitable chemical medium controlled by a robotic hand: closed loop experiments. *Int. J. Bifurcat.Chaos* **14**:3347–3354 2004

[Zhang & Adamatzky 2008] L Zhang, A Adamatzky. Implementation of two-bit adder in excitable cellular automaton. *Chaos, Solitons & Fractals* 2008