Journeys in Non-Classical Computation

a UK Grand Challenge in Computing Research

Susan Stepney

Non-Standard Computation Group
Department of Computer Science
University of York

April 2006

The UK Grand Challenges in Computing

- UK Computing Research Committee (UKCRC) initiative
  - to discuss opportunities for advancement of computing science
  - (Nov 2002) original call resulted in 109 submissions, merged and refined into seven "Grand Challenges"

    http://www.ukcrc.org.uk/grand_challenges/index.cfm/

1. In Vivo -- In Silico : Andrew Bangham

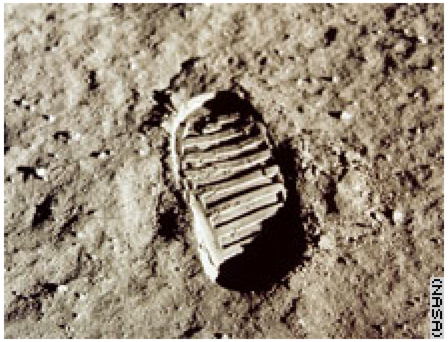
   The Worm, the Weed, and the Bug : breathing life into biological data

2/4. Global Ubiquitous Computing: Science & Design : Morris Sloman

3. Memories for Life : Nigel Shadbolt
5. Architecture of Brain and Mind : Murray Shanahan
6. Dependable Systems Evolution : Jim Woodcock
7. Journeys in Non-Classical Computation : Susan Stepney

   Robust, adaptable, powerful computation, as inspired by Nature

why "Journeys"?



http://www.cnn.com/2004/TECH/space/07/16/moon.landing/

- choosing the right metaphor
  - "goal" -- eg : proving whether P = NP
    - know where you are going
    - halting at the end-point
  - "journey" -- eg : Grand Tour of Europe
    - importance of entire process, not just the destination
    - exploration, open-ended, non-halting, …

- contributors to the Challenge statement, so far:
  - Susan Stepney, Samuel L. Braunstein, John A. Clark, Andy Tyrrell UYork
  - Andrew Adamatzky, Robert E. Smith UWE
  - Tom Addis UPortsmouth
  - Colin G. Johnson, Jonathan Timmis, Peter Welch UKent
  - Robin Milner UCambridge
  - Derek Partridge UExeter

http://www.cs.york.ac.uk/nature/gc7/

Classical computation assumptions

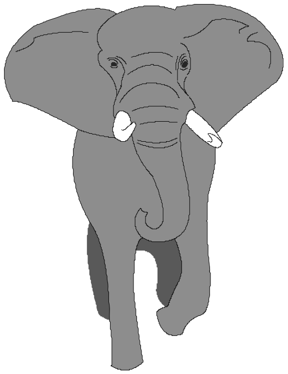
- Turing paradigm
  - finite discrete classical state machine, Halting, Universal
  - closed system, predefined state space
- Von Neumann paradigm
  - sequential fetch-execute-store
- algorithmic paradigm
  - initial input … deterministic function … final output
  - black-box isolated from the world
- refinement paradigm
  - a known specification is refined to provably correct code
- pure logic paradigm
  - substrate (hardware/physics) is irrelevant

Non-classical views

- Real World as inspiration
  - natural computation : physics-inspired, bio-inspired
  - massive parallelism
    - emergence, "more is different"
- Real World as a computer
  - all computation and all data is ***embodied***
    - physical effects – particularly quantum
  - analogue computation
      - the great missed opportunity of the 20th Century?
    - eg, protein folding problem
- Open dynamical systems
  - no Halting, rather ongoing developing interactive processes
    - computation itself as a journey, not a goal

"non-classical" computation?

like defining the bulk of zoology by calling it the study of 'non-elephant animals'



– Stan Ulam (attrib) on the name "non-linear science"

non-linear science / non-classical computation

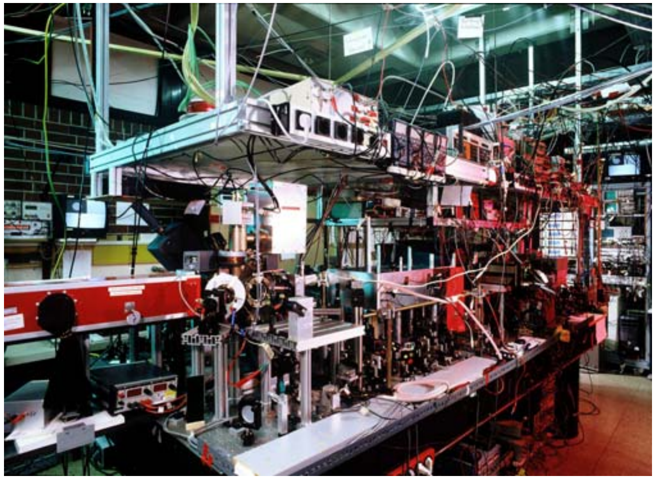
Here be Dragons

linear science / classical computation

Journeys suggested so far

- **Quantum Software Engineering**
  - computing with weird physics
- Reaction-diffusion and excitable processors
  - computing with spatio-temporal chemistry
- **Artificial Immune Systems**
  - computing with biology, and its metaphors
- Evolvable hardware
  - hardware that can adapt, evolve, grow, repair, replicate, learn, …
- Approximate Computation
  - Non-boolean: statistics and probabilities
- Socially sensitive computing
  - eschewing the philosophy of crispy defined sets for Wittgensteinian "families"
- **Open Dynamical Systems**
  - far-from-eqb, phase spaces, trajectories and attractors
- **Molecular Nanotechnology**
  - preparing for the Diamond Age
- **Massive Concurrency**
  - CSP + π-calculus + execution = occam-π

Quantum Software Engineering



http://heart-c704.uibk.ac.at/recent/teleportation/quantum_computer_experiment_1.jpg

- ***why Quantum?***
- exciting new capabilities
  - superposition
    - exponential speedup of (some) algorithms
  - *non*-pseudo random numbers
  - entanglement
    - secure cryptography ("no cloning"; teleportation)
  - …

quantum computational models

- how do classical models generalise to quantum domain?
  - classically -- many different but equivalent formalisms
    - from Turing machines to fixed point approaches
  - quantum analogues -- may not be so clearly equivalent
    - quantum superposition of *all* fixed points?
  - maybe some generalisations are more appropriate than others
    - but today, focussed mainly on the circuit model
    - need to tackle other models
  - we do know that we *don't* know all the consequences of the quantum world
    - new advances in entanglement, quantum information, …
- exploring these various generalisations may also give deeper understanding of their classical counterparts

weird science

- quantum computing makes it indisputable that theories of computation depend crucially on the laws of ***physics***
  - computation is **embodied** in the physical world ; it is not merely a *mathematical* abstraction

"Turing thought that he understood paper. But he was mistaken."

[Feynman (attrib by Deutsch)]

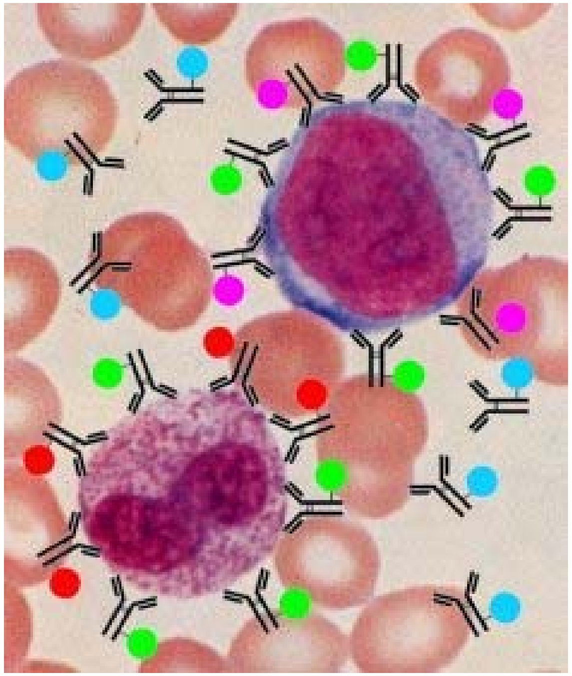
- what *other* laws of (non-classical) physics can we exploit for computation?
  - special and general relativity ; string theory ; branes ; …

Challenge: a theory of computation embodied in different physical paradigms

Artificial Immune Systems

- ***why AIS?***

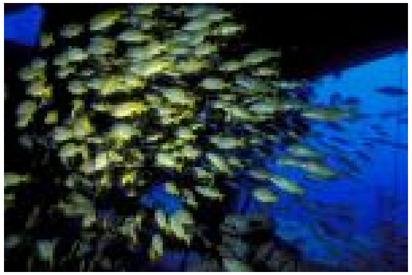
- bio-inspiration, from a complex (and complicated!) biological system
- several biological models, inspiring different computational metaphors



http://www.hmds.org.uk/mabs.html

AIS : computational models

- non-classical bio-inspired algorithms
  - how to exploit essential non-deterministic / stochastic nature
  - how to design, build and use a ***continually learning*** system
    - the real immune system has no final output, does not Halt
- non-classical refinement



  - how do global classifiers and recognisers *emerge* from low level non-specific agents
    - how to design rigorously (if non-incrementally) desired emergence
    - how to reason rigorously, about use in critical applications

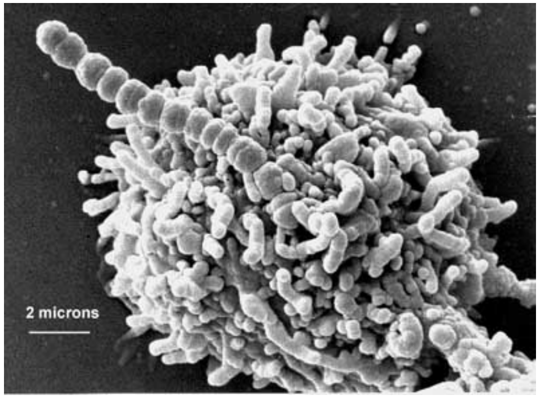
Challenge: a unified theory of learning systems

with evolutionary, neural, immune, … as special cases

AIS : wet computation

- the real immune system is vastly more complicated than our current computational metaphors
  - can we extract more realistic, but still useful, computational concepts and metaphors from the real immune system?

- can we compute using components from the real immune system?
  - DNA computing uses real physical wet DNA



http://www.ncl.ac.uk/facilities/microscopy/tcell.htm

Challenge: computation with agents from real biological, chemical, physical systems

AIS : embodiment

- what is the effect of the physical substrate on the workings of the real immune system?
  - can all immune responses be implemented on *any* substrate?
  - if not, what do "alternative immune systems" look like on alternative substrates?
  - how can we theoretically unify these alternative systems?

- do diseases exploit the immune substrate?
  - do diseases exploit the system's computational limitations?

Challenge: a theory of the effect of the given substrate on any biological system

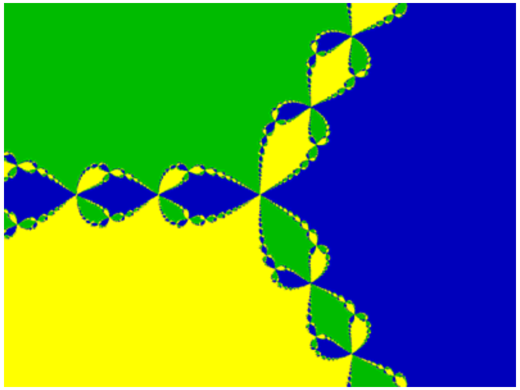
Biological necessities

- we see many features in biology
  - but have only *one* exemplar : terrestrial life
- what are necessary for any complex adaptive system?
  - necessary for adaptability, robustness, …
- what are necessary on the given substrate that implements the system?
  - carbon *versus* silicon necessities
- what parts are merely contingent evolutionary aspects?
  - different if "the tape were played again"

Challenge: unified theory of biological computation

"better than reality", "different from reality" systems

Open Dynamical Systems

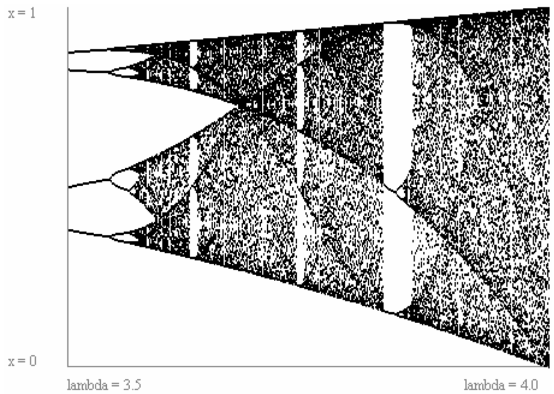


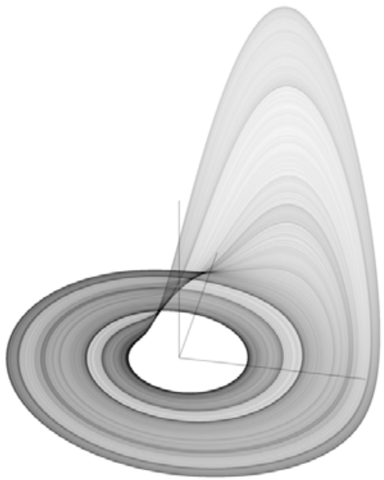
http://www.mcgoodwin.net/julia/MCMNEWT.GIF

- ***why Dynamical Systems?***
- many complex biological processes
  appear to have an underlying dynamical systems structure
- emergent properties are related to underlying dynamics
- phase space and attractors yield a suggestive computational metaphor

trajectories and attractors

- **computation** emerges as (an observed projection of) the trajectory through the computational space
    - (in the case of fully dynamic computation with no halting)
- the trajectory is governed by the structure of the underlying state/phase space and its ***attractors***
  - the state/phase space itself may be dynamic
    - parameters changing due to environment, etc





http://commons.wikimedia.org/wiki/Image:Roessler_attractor.png

trajectories as computations?

- can a computation be expressed as a trajectory in a phase space of attractors
  - with the attractors affected by parameters / inputs?
  - design for robustness: large basins of attractions?
- can a computation be expressed as movement between the unstable periodic orbits of a strange attractor?
  - how are these orbits selected?
- how can we reason about "open" phase spaces?
  - where the phase space changes *in kind* as the computation progresses

Challenge: a new computational paradigm expressed in dynamical terms of attractors and trajectories

open, far-from-equilibrium systems

- **open** systems : constant addition of new resources
  - energy, matter, information flowing into and out of the system
- ***far-from-equilibrium*** systems
  - not in a steady state, maybe at computational "edge of chaos"
    - structure "on all scales"
- tend to form stable structures, ***patterns***, that persist
  - *stable*, but not *static*
    - can change readily in response to stimuli, are "*poised*"
  - these higher level structures (patterns, agents) in space and time are ***emergent properties***

emergence : "more is different"

- **emergence** : the difference between a **mere aggregation** of component parts, and a **complex *system***
  - "more than the sum of its parts"
    - pile of bones v. skeleton
    - homogenised "hamster in a blender" v. living organism
      - homogenisation keeps the **components**, but removes all the **connections** and **relationships**
- **high level systems *emerge*** as **patterns of structure and behaviour** from structure and dynamics of the simpler rules governing **underlying agents**
  - emergent properties are described in new terms
    - ***emergent properties*** may be the names we give to trajectories on complex attractors

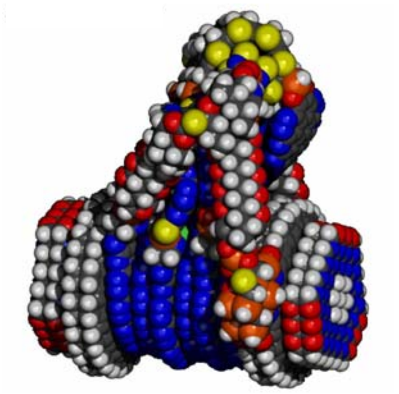
hierarchies of emergence

- the *stable emergent structures* in space and time have their *own* structure and dynamics
  - their own phase spaces, trajectories, and attractors
  - still higher level patterns can then emerge from this new dynamical phase space
    - molecules emerge from atoms ; cells emerge from molecules ; tissues from cells ; organs from tissues ; ...
    - **life** emerges from matter with ***structure and dynamics***
      - life as a structured, dynamical ***process*** (not as a static "thing")
  - ***hierarchical emergence*** is essential for **non-trivial** systems
    - challenge of pre-defining the emergent phase space?

Challenge: a unified theory of open dynamical systems

Molecular nanotecnology

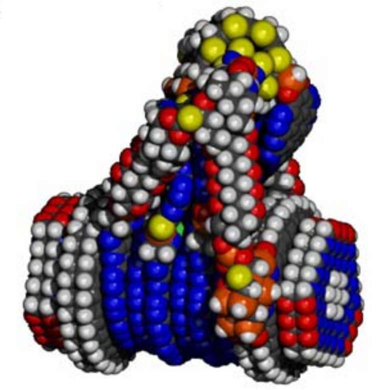
- ***why MNT?***



- potentially staggering technology
- example of embodied emergence
- needs Non-Classical Computational results

nanotechnology: “molecular manufacturing“

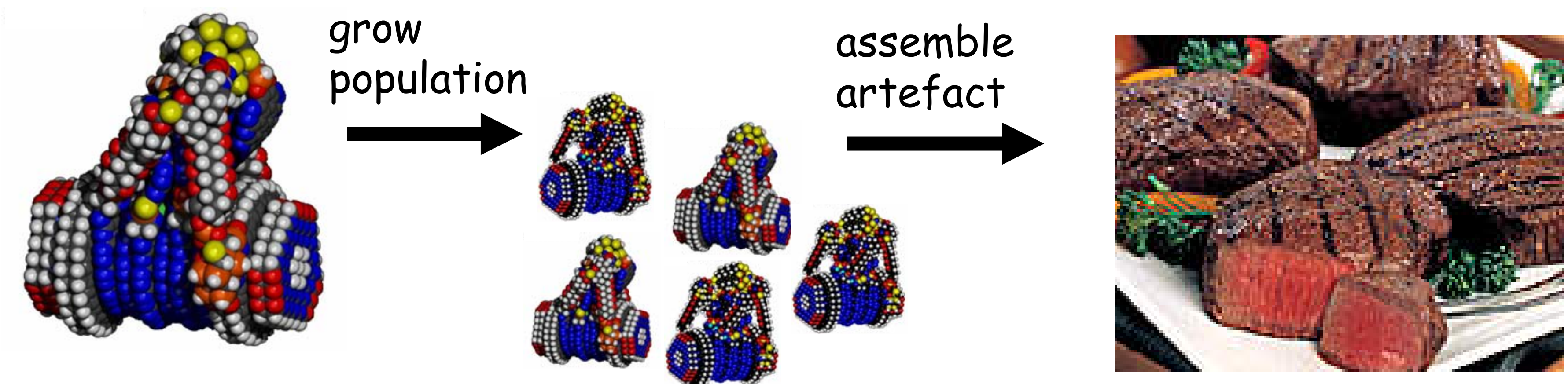
- ***molecular nanotechnology (MNT)*** [K. Eric Drexler, 1986, 1992]
  - molecular scale programmable “robots”
    - “assemblers”, “nanobots”, “nanites”, ...
  - mechanically positioning reactive molecules
  - making *macroscopic* artefacts
    - *not* talking about “macroscopic” nanotechnology
      - electron microscopes etc moving atoms around, making microscopic artefacts



- universal assembler
  - given the right raw materials, and the ***right assembly instructions***, can assemble *anything*, from steaks to spaceships
    - leaving hardware/wetware to physicists, engineers, biologists, ...
    - strange physics at very small sizes
      - effects of friction, flow, gravity, etc all very different

assembling artefacts

- growth and development on two levels
  - bootstrap a small initial assembler population
    - pool of raw material (mainly carbon)
    - assemble trillions of nanites (exponential growth)
  - eg, to build a new nano-fabrication plant
    - which then assembles, or "grows", the artefact



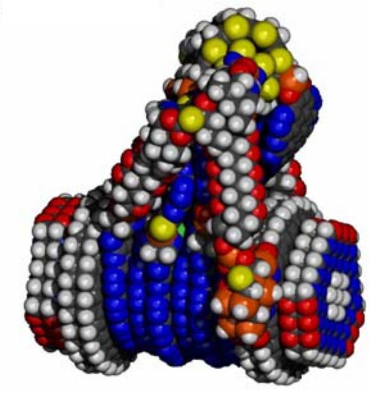
http://www.imm.org/

http://www.omahasteaks.com/

the MNT *design* challenge

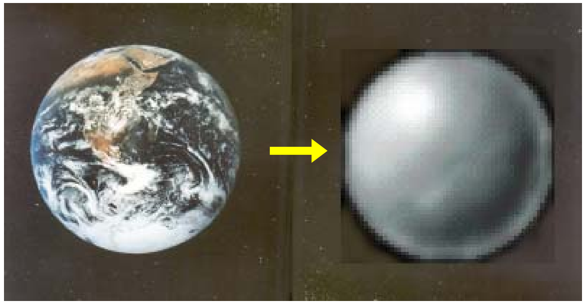
- assembled artefact is ***emergent property***
  - of actions of vast number of nanites
- design requires "reverse emergence"
  - from desired emergent artefact
  - to behaviour of nanite assemblers
    - extreme example of "non-classical refinement"
    - simple rules give complex behaviour
      - but *which* simple rules give the *desired* complex behaviour?

design appropriate assemblers →



the MNT *safety* challenge

- given vast numbers of nanites, some will go wrong
  - if they are self-replicating, they will evolve
    - evolution is an inevitable consequence of "reproduction, variation, selection"



- safety critical application
  - "**Grey Goo**" scenario [Global Ecophagy]
  - current approaches totally inadequate
    - "proof of correctness" doesn't help with a mutant
      - the mutant is a *different system*
  - new safety techniques and tools required
    - *design* of non-viable "adjacent possible"
    - evolution will exploit *anything*
      - even (especially) things outside your abstract model
      - particularly the *embodied properties* of the substrate

Engineering emergence

- classic approaches to SW Engineering …
  - formal specification ; *design by refinement* ; total correctness

  … don't work with emergent properties

  F. Polack, S. Stepney. Emergent Properties do not Refine. *REFINE 2005*, ENTCS

- non-classical techniques required
  - phase spaces, dynamics, trajectories, attractors, …
  - probabilistic and "soft" reasoning
  - (initial) understanding via ***patterns*** ?
    - patterns of structure and dynamics

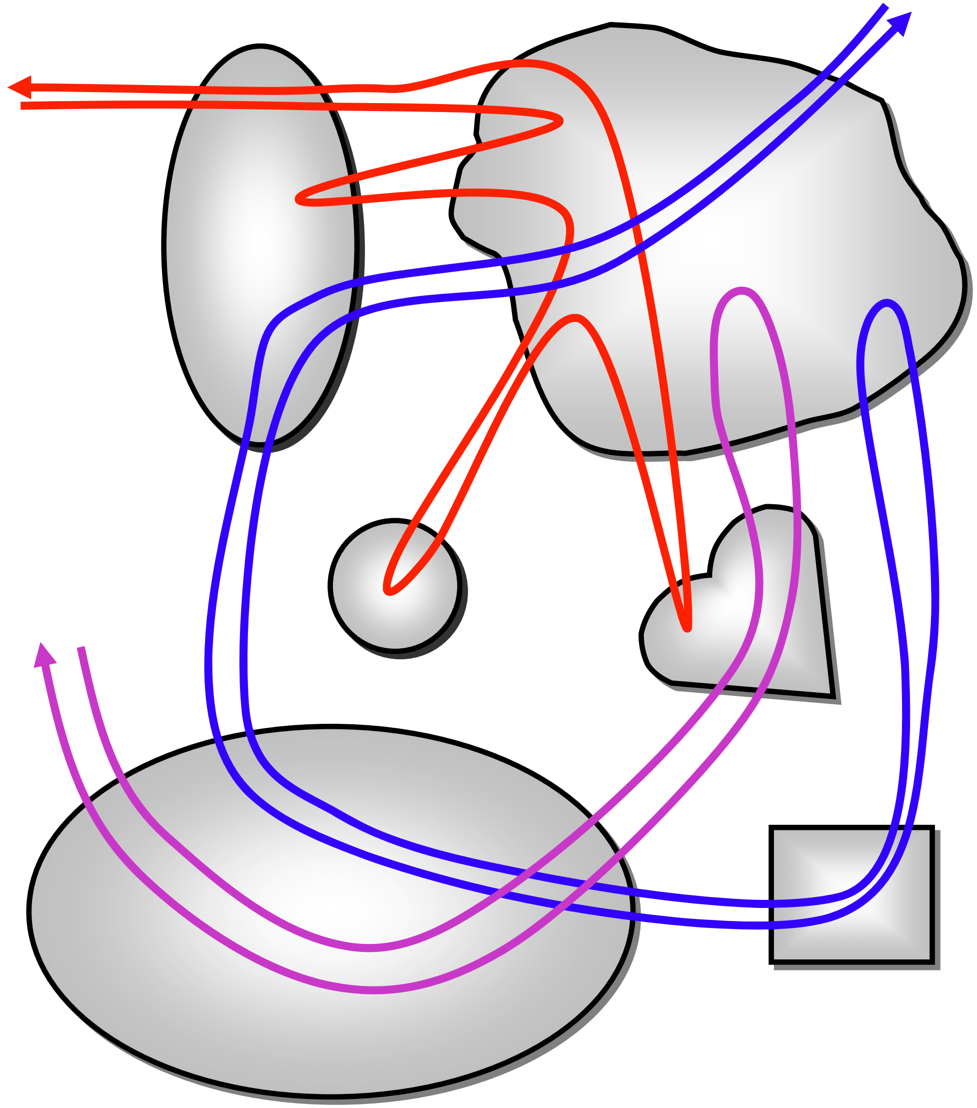
Challenge: to engineer emergent properties of embodied computational agent systems (nanites!)

Massive concurrency

- ***why concurrency?***
- the real world is *massively* parallel
  - with no central point of control
- our traditional concurrent programming paradigms are clumsy
  - (a few dozen) threads, etc
    - we take an intrinsically parallel world, sequentialise it, then add the wrong sort of concurrency back on

Thread spaghetti [Peter Welch]

Each single thread of control snakes around objects in the system, bringing them to life ***transiently*** as their methods are executed.



Threads cut across object boundaries leaving spaghetti-like trails, ***paying no regard to the underlying structure***.

CSP + π-calculus + execution = occam-π

- CSP as a model of concurrency and communication
  - remember occam on the transputer?
    - Handel-C for FPGAs
  - clean semantic model, but restricted
    - static, fixed, pre-defined processes and channels
- π-calculus for mobility
  - of both channels and processes
- Peter Welch's group at Kent
  - lovely implementations, including JCSP, and **occam-π**
  - proposed as a testbed/infrastructure for GC-7 work

concurrency as a fundamental computational paradigm

a clean and simple conceptual model; an efficient implementation

The Grand Challenge

to produce a fully mature science of all forms of computation, that embraces the classical and the non-classical paradigms

- many journeys, one Challenge
- like all science, the Challenge is an ongoing journey

http://www.cs.york.ac.uk/nature/gc7/