A physical/embodied/open/emergent perspective on computing

Susan Stepney, University of York

Physical

Classical computation purports to assume a mathematical view of computation, independent of the laws of physics.

Yet all computation is performed by some physical device. Physics matters. The Turing model is actually implicitly a Newtonian model: (i) each place on the tape can have only a single symbol written on it, whereas quantum physics allows a *superposition* of multiple symbols; (ii) the observer and the computation are co-located, or at least experience the same proper time, whereas general relativity allows significantly different proper times (eg finite versus infinite) to elapse in different locations for different observers. This implicit Newtonian model means that *the entirety of classical computation is based on a demonstrably false premise*.

Embodied

Classical computation assumes a disembodied, “black box” view of computation, with no interaction with the environment between initial input and halting.

Yet the majority of computational system can sense and manipulate their environment as they compute (from a minimal touch screen interface, through embedded controllers with sensors and actuators, to fully embodied mobile robots). The system’s internal state depends on what it senses as well as what it computes, and its manipulations depend on this state. The manipulations change the environment, and hence potentially change what is subsequently sensed, and hence changes the system’s subsequent state and its further manipulations. This produces a complex dynamical feedback process: the system is *embodied* in its environment. Moreover, the environment provides a rich and active resource, which can be exploited to take up some of the computational burden of the system (for example, memory via stigmergy, or more computational via *in materio* processing).

Open:

Classical computation assumes a closed system: there is a predefined requirement specification (in principle, if not always in practice!), and a predetermined and fixed state space (in that the relevant Turing machine’s tape symbol set and its state transition function are defined and fixed before the computation starts).

Yet embodied computation interacts with a complex rich dynamic environment. That environment is itself *open* and far from equilibrium, exhibiting *continual novelty*, including an evolving, growing, changing state space. No matter how well-modelled the environment might be, there are inevitably aspects of the environment not captured by the model: those aspects may nevertheless impinge on the computation. Computation in an open environment cannot be predefined, and must adapt to new situations, and to new kinds of situations.

Emergent:

Classical computation looks to a top-down development process based on the theory of refinement, where a crisp well-defined mapping exists between the high level specification and the low level implementation.

Yet complex systems exhibit emergent properties, where no such well-defined mapping exists. The high level description and the low level realisation are expressed with different concepts, in different languages. In particular, emergence seems to require a “process” view, rather than a “substance” view: entities comprised of *patterns of behaviour*, not “stuff”. We are not as good at thinking and modelling in terms of processes. For example, think of UML models: they usually have copious class diagrams and very few, if any, sequence diagrams (although there may be a variety of reasons for this). Additionally, a process view allows entities that are independent of the details of the underlying substrate, meaning that the same processes could be supported by different substrates.

Taking a physical, embodied, open, emergent view of computation will lead to a radical new way of approaching these systems, one that acknowledges the importance of the dynamic coupling with an open, continuously novel environment. Such an approach is a significant research challenge.

(self) refs:

- Susan Stepney. The Neglected Pillar of Material Computation. (submitted)
- Susan Stepney. Embodiment. In Flower & Timmis, eds. *In Silico Immunology,* chapter 12, pp 265-288, Springer, 2007
- Susan Stepney, Fiona Polack, Heather Turner. Engineering Emergence. *ICECCS 2006: 11th IEEE International Conference on Engineering of Complex Computer Systems, Stanford, CA, USA, August 2006.* IEEE, 2006
- Susan Stepney, Samuel L. Braunstein, John A. Clark, Andy Tyrrell, Andrew Adamatzky, Robert E. Smith, Tom Addis, Colin Johnson, Jonathan Timmis, Peter Welch, Robin Milner, Derek Partridge. Journeys in Non-Classical Computation II: Initial journeys and waypoints. *Int. J. Parallel, Emergent and Distributed Systems.* **21**(2):97-125, April 2006
- Susan Stepney, Samuel L. Braunstein, John A. Clark, Andy Tyrrell, Andrew Adamatzky, Robert E. Smith, Tom Addis, Colin Johnson, Jonathan Timmis, Peter Welch, Robin Milner, Derek 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