Programs as tools for knowledge

Henri Salha, IHPST – Paris-I University

HAPOP4 symposium
Oxford, 23rd March, 2018

What can we learn from computer programs?

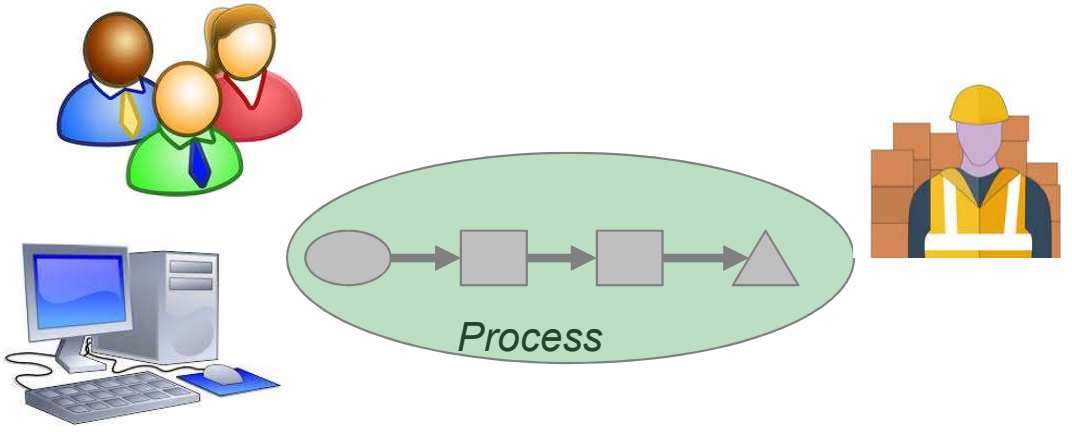
Some preliminary definitions

Scope restriction: programs at run-time,

i.e. computational processes

A computational process is either (1) the execution of a rule-based sequence of pre-defined operations, by a digital computer, involving the handling of data or exchange of data with an external context;
or (2) the rule-based parallel execution of many computational processes.

What is the context of a process?



Why this question? Cognitive sciences, computational science

Dennett 1980:

"AI [shares] with traditional epistemology […] the top-down question: how is knowledge possible? […] It shares with psychology in distinction from philosophy a typical tactic [which is to] ask themselves an easier preliminary question:

"How could any system (with features A, B, C, . . .) possibly accomplish X ?"

This is an engineering question, a quest for a solution (any solution) rather than a discovery"

Humphreys 2004:

Computational methods now play a central role in the development of many physical and life sciences. […] […] [These] developments, which began in the 1940s and accelerated rapidly in the last two decades of the 20th century, have given rise to a new kind of scientific method that I shall call computational science. (p49)

Computer simulations (1)

A core topic of epistemology, with well known problems

- Empirical vs. a priori: status of results as theoretical extrapolations or genuine observations
- Justification ("Simulations are only as good as their assumptions")
- Epistemic opacity: practical inability of the epistemic agent to check the computations

… But also an issue to define which programs should count as simulations

- How to distinguish simulations from simple numerical calculations programs?

Humphreys 2004, p112

In the early days of computer simulations, static numerical arrays were all that was available, and it would seem unreasonable to disallow these pioneering efforts as simulations. […].

The definitions we have given up to this point might seem to give us no reason to claim that computer simulations are essentially different from methods of numerical mathematics.

Computer simulations (2)

S. Hartmann, 1996, p.82

« *A simulation imitates one process by another process*. In this definition, the term “process” refers solely to some object or system whose state changes in time. »

1) Imitation

2) Time

About the time dimension of programs: two broad families

Manna & Pnueli, 1992, p.3

*" **A transformational program** is the more conventional type of program, whose role is to produce a final result at the end of a terminating computation. Consequently, the useful view of a transformational program is to consider it as a […] function from an initial state to a final state or a final result. […] For such specifications, ordinary predicate logic provides an adequate formulation and reasoning tool.*

***The role of a reactive program**, on the other hand, is not to produce a final result but to maintain some ongoing interaction with its environment. Examples of reactive programs are operating systems and programs controlling mechanical or chemical processes, such as a plane or a nuclear reactor. […] They cannot be specified by a relation between initial and final states, [... It is for] such programs that the formalism of temporal logic […] is recommended. "*

See also Lehman 1980, distinction between S & E programs is somewhat similar

Epistemic power of functional programs

Data analysis
(aka Knowledge Discovery from Databases)

Archiving & communication

Data organization & visualization

Finding, grouping & sorting

Classification & Regression

Data description (eg clustering)

Pattern recognition

…

Analytical developments

Translations & transcoding

Maths analysis

Combinations & developments

(eg tree exploration)

Iterations and recursion

…

Epistemic power of reactive systems

Examples

Embedded systems, production systems, robots

Enterprise software (workflow management)

Knowledge in action – “know-how”, ability

But only indirect knowledge for the observer

- Data extraction for functional analysis
- Performance analysis, learning from experiment

No need of formal semantics

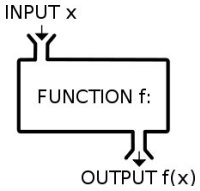
Brooks, 1991:

Even at a local level we do not have traditional AI representations. We never use tokens which have any semantics that can be attached to them. The best that can be said in our implementation is that one number is passed from a process to another. [...] To a large extent the state of the world determines the action [and not a model of the world]

Two interpretations of knowledge map broadly with our two families of computations

	"Know-that" Symbolic knowledge	"Know-how" Practical knowledge
Output converts to…	Representation	Action
Output values to…	Truth	Success
Computation means…	Relationships (laws)	Behaviors
Context	Abstracted (black-boxing)	Situated (interactive)
Epistemic warrant	Semantic model	Feedback mechanism
Thought Traditions	Functional cognitivists Philosophers of Mind	Embodied cognitivists Pragmatists / constructivists

The scale of concurrency and openness: any intermediate cases?



Pure functional programs

Pure reactive systems

- *Sequential*
- *No context*(1)

- *Concurrent*
- *Open context*

(1) Except at start-up, as inputs

Intermediate cases?

Testing / training reactive systems in controlled environments

Aiming to shut down any unexpected event

Other examples: tests of new software through mass data feeds

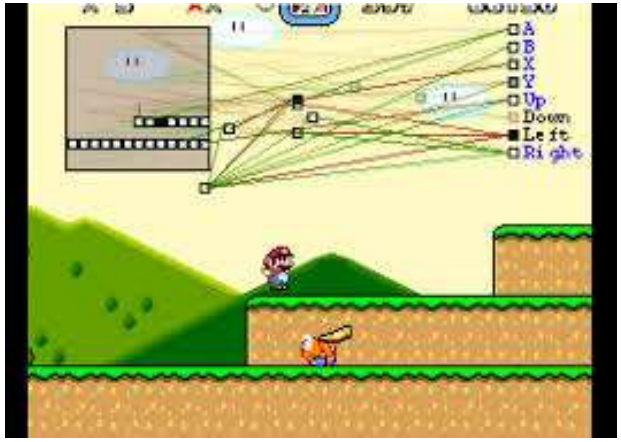
Testing and training users in controlled environments

Part of the context is data fed (e.g. the travel data), part of the context is object of the test (e.g. the apprentice-pilot)

Games are the general form of « closed context » processes

Concurrency is still very much alive, but the computational process is embedded in a context which is ideally shut down from reality: gamers « play a role » with pre-defined moves

Further cases



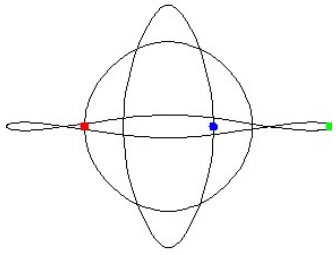
The logic can continue and a given context can always be modelled to be embedded in a larger process

Illustration: Neuronal Network trained to play and win Super-Mario

- The gamer is emulated by another computational process interacting with the game process
- Together they form a new concurrent process, with no external active context

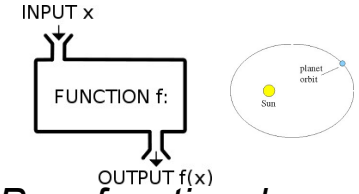
« Zero-player games » are pure cases which are both functional (no context) and concurrent – all the context has been virtualized

- Illustration: Robotwar, first programming game where the players' interventions take place before the actual battle begins
- Game of life, one of the first agent-based simulations



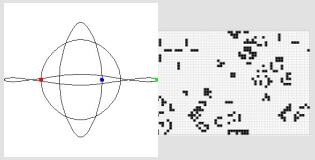
Other multi-variable numerical iterations, such as Euler method for simulating the 3-body problem, taking the form of pseudo-concurrency (round-based updates of the bodies cinematics) may be interpreted as « games » in this latest sense

The scale of concurrency and openness



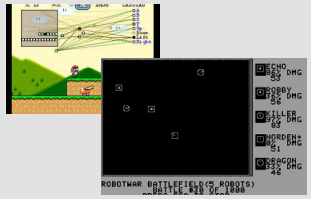
Pure functional programs
Simple variable iterative computation

- *Sequential*
- *No context*



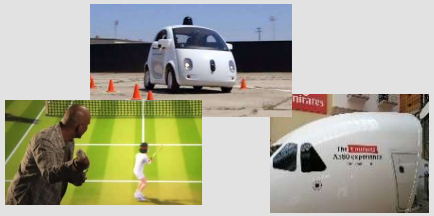
Agent-based models
Interdependent variables iterative computations

- *Pseudo-concurrent*
- *No context*



Video-game + Emulator
Agent-based models

- *Concurrent*
- *No context*



Tests of reactive systems
Video-Games
Simulators

- *Concurrent*
- *Closed context*

Pure reactive systems

- *Concurrent*
- *Open context*

« Simulations »

« Games »

Conclusions

Three categories of computer programs which show different uses in knowledge endeavors

- Functional programs covering analytical functions (incl. induction)
- Reactive programs covering technical and practical functions
- Games at the interplay between these two sides of knowledge

Could this categorization prove helpful to the debate between functional cognitivists and dynamical / situated cognitivists?

Other leads for exploration:

- Epistemological issues linked to simuations
- Ontology of computational processes?
- Epistemology of programming

Bibliography (1)

Barberousse, Anouk, Sara Franceschelli, and Cyrille Imbert, 'Computer Simulations as Experiments', *Synthese*, 169 (2009), 557–74

Barberousse, Anouk, and Marion Vorms, 'About the Warrants of Computer-Based Empirical Knowledge', *Synthese*, 191 (2014), 3595–3620

Beisbart, Claus, 'How Can Computer Simulations Produce New Knowledge?', *European Journal for Philosophy of Science*, 2 (2012), 395–434

Brooks, Frederick P., *The Mythical Man-Month: Essays on Software Engineering* (Boston, Mass.: Addison Wesley, 1982)

Brooks, Rodney A., 'Intelligence Without Reason', in *Proceedings of the 12th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'91 (San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1991), pp. 569–595———, 'Intelligence without Representation', *Artificial Intelligence*, 47 (1991), 139–59

Burge, Tyler, 'Computer Proof, Apriori Knowledge, and Other Minds', *Noûs*, 32 (1998), 1–37

Cartwright, Nancy, *How the Laws of Physics Lie* (Oxford New York: Clarendon Press Oxford University Press, 1983)

Clark, Andy, *Mindware: An Introduction to the Philosophy of Cognitive Science*, 2 edition (New York: Oxford University Press, 2013)

De Mol, Liesbeth, *Looking for Busy Beavers. A Socio-Philosophical Study of a Computer-Assisted Proof* (College Publications, 2012)

Dennett, Daniel C., *Brainstorms: Philosophical Essays on Mind and Psychology*, Fortieth Anniversary Edition (Cambridge, MA: MIT Press, 2017)

Dowek, Gilles, *Les Métamorphoses du calcul* (Humensis, 2015)

Dupré, John, 'A Process Ontology for Biology', *The Philosophers' Magazine*, 2014, 81–88

Eden, Amnon H., 'Three Paradigms of Computer Science', *Minds and Machines*, 17 (2007), 135–67

Floridi, Luciano, 'Semantic Conceptions of Information', in *The Stanford Encyclopedia of Philosophy*, ed. by Edward N. Zalta, Spring 2017 (Metaphysics Research Lab, Stanford University, 2017)

Bibliography (2)

Fodor, Jerry A., *The Language of Thought* (Cambridge, Mass: Harvard University Press, 1980)

Hartmann, Stephan, 'The World as a Process', in *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*, Theory and Decision Library (Springer, Dordrecht, 1996), pp. 77–100

Hughes, Richard IG, 'Models and Representation', *Philosophy of Science*, 64 (1997), S325–S336

Humphreys, Paul, *Extending Ourselves: Computational Science, Empiricism, and Scientific Method*, New Ed (Oxford: OUP, 2007)

———, 'The Philosophical Novelty of Computer Simulation Methods', *Synthese*, 169 (2009), 615–26

Kitchin, Rob, 'Big Data, New Epistemologies and Paradigm Shifts', *Big Data & Society*, 1 (2014), 2053951714528481

Lehman, M. M., 'Programs, Life Cycles, and Laws of Software Evolution', *Proceedings of the IEEE*, 68 (1980), 1060–76

Manna, Zohar, and Amir Pnueli, *The Temporal Logic of Reactive and Concurrent Systems - Specification*, 2 vols (New York: Springer-Verlag, 1992), I

McConnell, Steve, *Code Complete: A Practical Handbook of Software Construction, Second Edition*, 2nd edition (Redmond, Wash: Microsoft Press, 2004)

McLaughlin, Brian P., 'Computationalism, Connectionism, and the Philosophy of Mind', in *The Blackwell Guide to the Philosophy of Computing and Information*, ed. by Luciano Floridi (Oxford, UK: Blackwell Publishing Ltd, 2003), pp. 135–51

Moor, James H., 'Three Myths of Computer Science', *The British Journal for the Philosophy of Science*, 29 (1978), 213–22

Morgan, Mary S., and Margaret Morrison, eds., *Models as Mediators: Perspectives on Natural and Social Science*, Ideas in Context, 52 (Cambridge: Cambridge University Press, 1999)

Norton, John D., 'Are Thought Experiments Just What You Thought?', *Canadian Journal of Philosophy*, 26 (1996), 333–66

Petricek, Tomas, 'Miscomputation in Software: Learning to Live with Errors', *The Art, Science, and Engineering of Programming*, 1 (2017)

Pias, Claus, 'On the Epistemology of Computer Simulation', *Zeitschrift Für Medien-Und Kulturforschung*, 2011 (2011), 29–54

Priestley, Mark, *A Science of Operations: Machines, Logic and the Invention of Programming*, History of Computing (London: Springer London, 2011)

Bibliography (3)

SethBling, *Marl/O - Machine Learning for Video Games* <https://www.youtube.com/watch?v=qv6UVOQ0F44> [accessed 19 December 2017]

Simon, Herbert A., *The Sciences of the Artificial*, 3. ed., [Nachdr.] (Cambridge, Mass.: MIT Press, 2008)

Soare, Robert I., 'Computability and Recursion', *Bulletin of Symbolic Logic*, 2 (1996), 284–321

Strachey, Christopher, 'Fundamental Concepts in Programming Languages', *Higher-Order and Symbolic Computation*, 13 (2000), 11–49

Tedre, Matti, and Peter J. Denning, 'The Long Quest for Computational Thinking', in *Proceedings of the 16th Koli Calling International Conference on Computing Education Research*, Koli Calling '16 (New York, NY, USA: ACM, 2016), pp. 120–129

Turner, Raymond, *Computable Models* (London: Springer London, 2009)

———, 'Programming Languages as Technical Artifacts', *Philosophy & Technology*, 27 (2014), 377–97

———, 'Specification', *Minds and Machines*, 21 (2011), 135–52

———, 'Understanding Programming Languages', *Minds and Machines*, 17 (2007), 203–16

Turner, Raymond, and Nicola Angius, 'The Philosophy of Computer Science', 2013

Varenne, Franck, and Marc Silberstein, eds., *Modéliser & simuler : Epistémologies et pratiques de la modélisation et de la simulation? 2 vols* (Paris: Editions Matériologiques, 2013-2014)

White, Graham, 'The Philosophy of Computer Languages', in *The Blackwell Guide to the Philosophy of Computing and Information*, ed. by Luciano Floridi (Oxford, UK: Blackwell Publishing Ltd, 2003), pp. 237–47

Wing, Jeannette M., 'Computational Thinking', *Communications of the ACM*, 49 (2006), 33–35

Winsberg, Eric, 'Computer Simulations in Science', in *The Stanford Encyclopedia of Philosophy*, ed. by Edward N. Zalta, Summer 2015 (Metaphysics Research Lab, Stanford University, 2015)

———, 'Simulated Experiments: Methodology for a Virtual World', *Philosophy of Science*, 70 (2003), 105–25