Alcom-FT Technical Report Series
ALCOMFT-TR-01-163

On the Power of Interactive Computing*

Jan van Leeuwen[1] Jiří Wiedermann[2]

[1] Department of Computer Science, Utrecht University,
Padualaan 14, 3584 CH Utrecht, the Netherlands.
[2] Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic.

Abstract. In a number of recent studies the question has arisen whether the familiar Church-Turing thesis is still adequate to capture the powers and limitations of modern computational systems. In this presentation we review two developments that may lead to an extension of the classical Turing machine paradigm: interactiveness, and global computing.

Computers will soon be part of all objects around us. It will enable forms of algorithmic intelligence and communication of unprecedented versality. The question arises whether the traditional notions in computability theory are still adequate to capture the powers and limitations of the new systems that are arising. In particular, some developments seem to challenge the view that computational systems are necessarily recursive. The present understandings of computation may transcend what is normally expressed by the Church-Turing thesis (see e.g. [4] and [12]). In this presentation we will explore some possible ingredients for extending the traditional computational models.

1 Interactive computing

Among the powerful computational notions in the large and in the small, is the notion of 'interaction'. The notion typically applies to systems that consist of many components that compute and communicate with each other and with some external 'environment'.

The purpose of an interactive system is usually not to compute some final result but to react to or interact with the environment in which the system is placed and to maintain a well-defined action-reaction behavior. Interactive systems are always operating and thus may be seen as machines on infinite strings, but differ in the sense that their inputs are not specified and may depend on intermediate outputs and external sources.

In the late nineteen seventies and early nineteen eighties, interactive (or: reactive) systems received much attention in the theory of concurrent processes (see e.g. [7, 8], [9]). Wegner [21, 22] recently called for a more computational view of reactive systems, claiming that they have a richer behavior than 'algorithms' as we know them. He writes ([22], p. 318):

> *"The intuition that computing corresponds to formal computability by Turing machines ... breaks down when the notion of what is computable is broadened to include interaction. Though Church's thesis is valid in the narrow sense that Turing machines express the behavior of algorithms, the broader assertion that algorithms precisely capture what can be computed is invalid."*

* This research was partially supported by GA ČR grant No. 201/98/0717 and by EC Contract IST-1999-14186 (Project ALCOM-FT).

The study of 'machines' working on infinite input streams (ω-words) is by no means new and has a sizeable literature, with the first studies dating back to the nineteen sixties and seventies (cf. Thomas [15], Staiger [14]). Nevertheless the notion of interactiveness adds a new perspective to it. For a discussion of Wegner's claims from the viewpoint of computability theory, see Prasse and Rittgen [10]). Formalizations of the theory of interaction are studied in Wegner and Goldin [23, 25], and Goldin [6].

In [18] we look at the computational implications of interaction. We give a simple model of interactive computing, consisting of one component C and an environment E interacting using single streams of input and output signals. The notion of 'component' that we use is very similar to that of a 'site machine' (see below). We identify a special condition, referred to as the interactiveness condition, which we impose throughout. Loosely speaking, the condition states that C is guaranteed to give some meaningful output within finite time any time after receiving a meaningful input from E and vice versa.

In the model we prove a number of general results for the interactive computing behaviour which a component C can exhibit, assuming that E can behave arbitrarily and unpredictably. We always assume that C is a program with unbounded memory, with a memory contents that is building up over time and that is never erased (unless the component explicitly does so). To understand the operation of interactive components, one may define a ω-string x to be interactively recognizable if C only outputs 1's (possibly interspersed with silent periods of finite duration) when it is fed x during its dialogue with E.

Example 1. The set $J = \{\alpha \in \{0,1\}^\omega | \alpha \text{ contains finitely many 1's}\}$ can be seen *not* to be interactively recognizable. Suppose there was an interactive component C that recognized J. Let E input 1's. By interactiveness C must generate a non-empty signal σ at some moment in time. E can now fool C as follows. If $\sigma = 0$, then let E switch to inputting 0's from this moment onward: it means that the resulting input belongs to J but C does not respond with all 1's. If $\sigma = 1$, then let E continue to input 1's. Possibly C outputs a few more 1's but there must come a moment that it outputs a 0. If it didn't then C would recognize the string $1^\omega \notin J$. As soon as C outputs a 0, let E switch to inputting 0's from this moment onward: it means that the resulting input still belongs to J but C does not recognize it properly. Contradiction.

Viewing components as interactive transducers of the signals that they receive from their environment one can show the following analogue to similar results from classical automata theory, using suitable definitions. The result is quite involved and heavily relies on the interactiveness condition.

Theorem 1. *A set $J \subseteq \{0,1\}^\omega$ is interactively recognizable if and only if it can be interactively generated.*

In [18] we also study a notion of interactively computable functions. We prove that interactively computable functions are limit-continuous, using a suitable extension of the notion of continuity known from the semantics of computable functions. We also prove an interesting inversion theorem which states that interactively computable 1-1 functions have interactively computable inverses.

2 Global computing

Among the new computational structures in the large, the Internet is beginning to play a very prominent role. Cardelli [3] has been among the first to realize the potential of the Internet as a programmable resource, suggesting that information structures may be realized on the Internet that can operate as 'global computers'. He foresees a multitude of (different) global computers operating on the Web, interacting with users and geographically distributed resources. The present developments in 'grid computing' (Foster and Kesselman [5]) are attempting to make this operational for the purposes of high performance computing.

Among the many questions raised by Cardelli [3], is the question what models of computation are appropriate for global computing. The Internet is an infrastructure for highly distributed forms of information exchange and computation, allowing large varying numbers of autonomous software entities to interact, compute and evolve under unpredictable circumstances. It is an environment in which the programs may depend on the data they have to operate on, the inputs are not given in advance and the computations may differ depending on the influence from unpredictable sources. This gives a computational power more akin to that of various 'non-uniform' computational models (cf. [1]), implying a power beyond that of ordinary Turing machines.

In [17] we describe a model of global computing in two stages. First 'site machines' are defined, by augmenting Turing machines with a communication facility. Next we define global Turing machines (or 'internet machines') as finite but time-varying sets of communicating site machines that can compute ad infinitum, modeling that a global computer may be evolving over time without limit. Under mild assumptions, the following theorem can be proved.

Theorem 2. *For every global Turing machine $\mathcal{M}$ there exists a Turing machine $\mathcal{T}$ with advice that sequentially realizes the same computations as $\mathcal{M}$ does, and vice versa.*

For an indication of the non-conventional power of Turing machines with advice we refer to Schöning [11] and Balcázar et al [1]. In [17] we also make a first step towards the development of a complexity theory for 'global space' and 'global time'.

3 Conclusion

Several developments in the theory of computation have challenged the kinds of computational models we are currently using (see [2], [13]). The given examples of interactive and global computing indicate that the classical Turing machine paradigm should be revised (extended) in order to capture the forms of computation that one observes in the systems and networks in modern information technology.

References

1. J.L. Balcázar, J. Diaz, and J. Gábarro. *Structural complexity*, Vol. I, 2nd edition, Springer-Verlag, Berlin, 1995.
2. L. Blum, F. Cucker, M. Shub and S. Smale. *Complexity of real computation*, Springer-Verlag, New York, NY, 1998.
3. L. Cardelli. Global computation, *ACM Sigplan Notices* 32-1 (1997) 66-68.
4. B.J. Copeland. The Church-Turing thesis, in: E.N. Zalta, *Stanford Encyclopedia of Philosophy*, Center for the Study of Language and Information (CSLI), Stanford University, Stanford, CA, http://plato.stanford.edu/entries/church-turing/, 1997.

5. I. Foster and C. Kesselman. *The grid: blueprint for a new computing infrastructure*, Morgan-Kaufmann Publ., San Francisco, 1998
6. D.Q. Goldin. Persistent Turing machines as a model of interactive computation, in: K-D. Schewe and B. Thalheim (Eds.), *Foundations of Information and Knowledge Systems*, Proc. First Int. Symposium (FoIKS 2000), Lecture Notes in Computer Science, vol. 1762, Springer-Verlag, Berlin, 2000, pp. 116-135.
7. R. Milner. *A calculus of communicating systems*, Lecture Notes in Computer Science, Vol. 92, Springer-Verlag, Berlin, 1980.
8. R. Milner. Elements of interaction, *C.ACM* 36:1 (1993) 78-89.
9. A. Pnueli. Specification and development of reactive systems, in: H.-J. Kugler (Ed.), *Information Processing 86*, Proceedings IFIP 10th World Computer Congress, Elsevier Science Publishers (North-Holland), Amsterdam, 1986, pp. 845-858.
10. M. Prasse, P. Rittgen. Why Church's Thesis still holds. Some notes on Peter Wegner's tracts on interaction and computability, *The Computer Journal* 41 (1998) 357-362.
11. U. Schöning. Complexity theory and interaction, in: R. Herken (Ed.), *The Universal Turing Machine - A Half-Century Survey*, Oxford University Press, Oxford, 1988, pp. 561-580.
12. J,.C. Shepherdson. Mechanisms for computing over arbitrary structures, in: R. Herken (Ed.), *The Universal Turing Machine - A Half-Century Survey*, Oxford University Press, Oxford, 1988, pp. 581-601.
13. H.T. Siegelmann. Computations beyond the Turing limit, *Science* 268 (1995) 545-548.
14. L. Staiger. ω-Languages, in: G. Rozenberg and A. Salomaa (Eds.), *Handbook of Formal Languages*, Vol. 3: *Beyond Words*, Chapter 6, Springer-Verlag, Berlin, 1997, pp. 339-387.
15. W. Thomas. Automata on infinite objects, in: J. van Leeuwen (Ed.), *Handbook of Theoretical Computer Science*, Vol. B: *Models and Semantics*, Elsevier Science Publishers, Amsterdam, 1990, pp. 135-191.
16. B.A. Trakhtenbrot. Automata and their interaction: definitional suggestions, in: G. Ciobanu and G. Păun (Eds.), *Fundamentals of Computation Theory*, Proc. 12th International Symposium (FCT'99), Lecture Notes in Computer Science, Vol. 1684, Springer-Verlag, Berlin, 1999, pp. 54-89.
17. J. van Leeuwen and J. Wiedermann. Breaking the Turing barrier: the case of the Internet, Techn. Rep. Institute of Computer Science, Academy of Sciences, Prague, 2000.
18. J. van Leeuwen and J. Wiedermann. A computational model of interaction, Techn. Rep. Dept. of Computer Science, Utrecht University, Utrecht, 2000.
19. P. Wegner. Interactive foundations of object-based programming, *IEEE Computer* 28:10 (1995) 70-72.
20. P. Wegner. Interaction as a basis for empirical computer science, *Comput. Surv.* 27 (1995) 45-48.
21. P. Wegner. Why interaction is more powerful than algorithms, *C.ACM* 40 (1997) 80-91.
22. P. Wegner. Interactive foundations of computing, *Theor. Comp. Sci.* 192 (1998) 315-351.
23. P. Wegner, D. Goldin. Coinductive models of finite computing agents, in: B. Jacobs and J. Rutten (Eds.), *CMCS'99-Coalgebraic Methods in Computer Science*, TCS: Electronic Notes in Theoretical Computer Science, Vol. 19, Elsevier, 1999.
24. P. Wegner, D. Goldin. Interaction as a framework for modeling, in: P. Chen *et al.* (Eds.), *Conceptual Modeling - Current Issues and Future Directions*, Lecture Notes in Computer Science, Vol. 1565, Springer-Verlag, Berlin, 1999, pp 243-257.
25. P. Wegner, D.Q. Goldin. Interaction, computability, and Church's thesis, *The Computer Journal* 1999 (to appear).