Analog Computational Power

I would like to comment on the report by Hava T. Siegelmann, "Computation beyond the Turing limit" (*1*). Siegelmann's thesis is that the natural class of efficiently computable functions for analog computation is P/poly, and thus that analog machines are potentially more powerful than conventional digital computers. The class P/poly is the class of functions that a Turing machine can compute in polynomial time with a polynomial amount of extra "advice." The number of bits of advice must be polynomial in the length of the input and may depend on this length, but not on the input itself. This class contains uncomputable functions such as the unary halting problem. The suggestion in Siegelmann's report is that a chaotic system, for example, Smale's horseshoe or the Baker's map, can obtain this extra advice and thus can compute the class P/poly. However, in a chaotic system, the extra advice is determined by the initial conditions, which one must presume are either determined by the computer programmer or are random. If they are determined by the programmer, the method for determining the advice could also presumably be used to supply it to a conventional computer. If the initial conditions are random, it seems that one could replace the advice by a random number generator and compute equally well. In this case, the class becomes BPP, which is generally considered to be the class of functions efficiently computable in practice.

The only other place that the advice used for P/poly might come from in an analog machine is a physical constant. However, the best current measurements of any physical constant are not accurate to much more than 10 decimal places, and this does not constitute a large amount of advice. Furthermore, unless the digits of a physical constant are in some way meaningful, they will be of no more help in solving problems than a random number generator.

The question Siegelmann raises is important. The issue of whether analog computation is more powerful than digital computation is a fundamental question that has not yet been adequately addressed; however, I find the arguments to this effect in Siegelmann's report unconvincing.

Peter Shor
AT&T Bell Laboratories
600 Mountain Avenue
Murray Hill, NJ 07974, USA

REFERENCES

1. H. T. Siegelmann, *Science* **268**, 545 (1995).

25 May 1995; accepted 14 September 1995

Over the years, many attempts have been made to define computational models with powers that exceed the Turing machine, as noted by Siegelmann (*1*), but these either rely on external information or on magical (non-Turing) modules that can, for example, answer questions that cannot be answered within the limits of the Turing machine. One simple example of this type of module is an "oracle," given the ability to perform a computation that cannot be performed by a Turing machine. One can easily construct an infinite hierarchy of computational power by positing the existence of oracles that perform progressively more complex computations that cannot be performed by machines that incorporate oracles of lesser power. For example, because the famous halting problem cannot be solved by a conventional Turing machine, a Turing machine with a Halting Problem Oracle is more powerful than a conventional Turing machine, because the oracle can answer the halting question. It is straightforward to define an unsolvable halting problem for the augmented machine, which simply leads one to define a more capable oracle to solve that problem. This construction can be continued indefinitely, yielding an infinite set of conceptual machines that are progressively more powerful than a Turing machine (*2*).

It is straightforward to define an oracle that provides the "advice" string discussed by Siegelmann. Thus, there does not seem to be any significant difference between the construction of machines that take advice and machines that incorporate oracle-like devices. And it is well known that these models do exceed the Turing machine's capability.

A major difficulty with the report by Siegelmann arises from her cavalier approach toward time and space limits. To begin, it is essential to the Turing model that there be no limit on time or space required to complete the computations, except that the time and space be finite so that the computation terminates. If one imposes more restrictive time or space limits on the computations, hierarchies of sub-Turing complexity can be constructed (*3*).

Siegelmann made an essential leap in the fifth paragraph of her report, where a polynomial limit was imposed on all computations. This immediately limits the discussion to a portion of the sub-Turing domain. Thus it is not unexpected that theorem 1 (*1*, p. 547) might be correct (though it is difficult to verify because the proof is not provided in the report, and the "sketch" does not address the complexity issues that must be an essential part of the proof). But the fact that one can compute, in polynomial time, something that was not of polynomial complexity in time under the Turing model does not mean that one has moved beyond the general Turing model, which has no time limits, except that the computation be finite.

A major difficulty with the report, and the reason for my commentary, is that the theorem and some other points do not violate the Turing Limit, as stated by Siegelmann. Through all of this, one must realize that these conceptual machine models are mathematical constructs that never can correspond to real machines, so this discussion has no practical significance.

Richard Y. Kain
Electrical Engineering Department,
University of Minnesota,
Minneapolis, MN 55455, USA

REFERENCES AND NOTES

1. H. T. Siegelmann, *Science* **268**, 545 (1995).
2. R. Y. Kain, *Automata Theory: Machines and Languages* (McGraw-Hill, New York, 1972), section 9.6.
3. J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation* (Addison-Wesley, Redwood City, CA, 1979).

5 June 1995; accepted 30 October 1995