THREE UNCONVENTIONAL COMPUTATIONAL PARADIGMS*

Selim G. Akl
School of Computing
Queen's University
Kingston, Ontario, Canada K7L 3N6

February 22, 2005

My recent work has uncovered at least three general unconventional computational paradigms within which parallel algorithms lead to a *superlinear* improvement in performance (that is, speed and quality) with respect to their sequential counterparts. These paradigms, described in some detail below, are: computing in real time, computing within the bounds of laws of nature, and computing subject to mathematical constraints. In some cases, the computations *cannot be performed* by any conventional computer (including the Turing machine). The growing presence of these new paradigms within the field of computing, and their increasing applications in society (with ubiquitous and mobile computing, for example, becoming more prevalent), lend timeliness and importance to these results. As well, it must be emphasized that each improvement in performance that I describe is consistent and provable, in the sense that it occurs in every instance of the computational problem under consideration. In addition, this improvement is independent of any discrepancies between the sequential and parallel computers used. These characteristics distinguish the results presented here from previous ones, where a superlinear speedup occurs only occasionally, or is achieved either because the sequential algorithm used is inefficient, or because the size of the memory on the sequential computer is restricted (an extensive list of references to such results is supplied in my book [1]).

1 Computing in the Presence of Real-Time Deadlines.

The notion of *real time* is an intuitive one. The term evokes simultaneity, liveness, and immediacy; it even conveys a sense of urgency and the need for an instantaneous action. Yet, *real-time computation* is difficult to capture. The numerous definitions in the literature are witness to this fact. One reason for this situation is that, in today's fast-paced society where computers can perform billions of operations per second, *every* computation seems to be performed in real time. Whether we are dealing with an automatic teller machine (ATM) or running a flight simulator, we have become accustomed to expect the result of our computations *now*. In a major textbook on the subject [7], the authors acknowledge the difficulty in delineating what computations to characterize as real-time ones. They proceed to define real-time computations as those computations, such as air traffic control (ATC), in which failure to produce the answers after a certain time would be catastrophic, involving loss of human life. This state of affairs provided the initial motivation for my work on a unifying definition of real-time computation that avoids the extremes just discussed. I sought a model that can be adapted to apply at any point on the spectrum (from ATM to ATC), depending on the circumstances and requirements of a particular application. I derived two such models; one uses a complexity-theoretic approach based on formal languages, the second uses state space traversal combined with discrete steepest descent. Most relevant in this context, however, were the algorithms that demonstrated superlinear performance improvement in several applications [3]. For example, I showed that for a computation involving a one-way function of n variables, an n-processor parallel algorithm leads to a speedup exponential in n. Similarly, a quality-up exponential in n is obtained when using n processors to find the zero of a continuous function numerically. In fact, in a cryptographic problem, the improvement in quality due to parallelism is *unbounded*, meaning that *no sequential computer succeeds in carrying out the required computation.*

*This research was supported by the Natural Sciences and Engineering Research Council of Canada.

2 Computing Under the Influence of Natural Laws.

Here my concern is with computations whose characteristics are akin to certain unique phenomena that are observed in different domains of science. The focus is on systems whose computations are subject to natural law—typically, systems whose variables are altered unpredictably whenever one of these variables is measured or modified. Examples of such computational environments include those in which *Heisenberg's uncertainty principle* of quantum physics is witnessed, or those in which *Le Châtelier's principle* of chemical systems under stress manifests itself. Our recent investigations have uncovered practical computations that are inherently parallel *in the strong sense*, meaning that they are efficiently executed in parallel, but impossible to carry out sequentially [6]. For example, we showed that a resistance-inductance-capacitance circuit (a linear dynamical system) undergoes significant perturbations that affect its dynamical behavior when some of its parameters (such as current, voltage, and so on) are measured sequentially. By contrast, these perturbations could be eliminated when the measurements are performed in parallel. Similarly, for the Belousov-Zhabotinskii chemical reaction (a nonlinear dynamical system), we showed that measurement disturbs the equilibrium of the system and causes it to go from a stable into an unstable state. If, however, several measurements are performed in parallel, the resulting perturbations cancel each other out and the system remains in a stable state. These results confirm an earlier existence proof, which I first gave in [2], regarding such computations. There it is shown that, given a system in equilibrium, the task of measuring the system's parameters, computing new values for them, and setting them to their new values before the system settles on its own into a new but undesirable state of equilibrium, *can be performed in parallel but not sequentially.*

3 Computing Subject to Mathematical Constraints.

In this final paradigm, the constraints are neither defined by the human operator (as in the first paradigm) nor dictated by the laws of nature (as in the second). Let A be a given physical object with a certain property p, and let B be another object also satisfying p but for which only a mathematical definition is available. It is required to construct B by applying a sequence of simple transformations to A such that each intermediate object (on the path from A to B) also satisfies p. For example, suppose that A and B are two grids, each in the shape of a triangulation (a planar graph all of whose faces are triangles) of a polygon with $2n$ vertices, as shown in Fig. 1 for $n = 4$. It is required to obtain B from A by applying a sequence of

Fig. 1

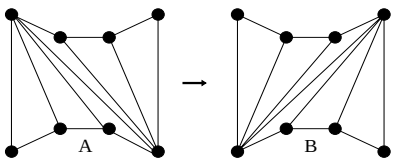
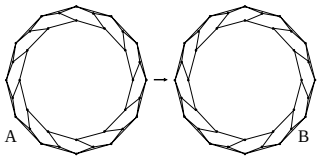


Fig. 2



transformations such that each intermediate grid is also a triangulation. A transformation is defined as the removal of at least one edge and the replacement of every removed edge by exactly one other. It is known that for triangulations with $2n$ vertices of the form A and B, respectively, $(n-1)^2$ transformations (each replacing one edge by another) are *required* by a sequential algorithm. In the special case where each pair of edges involved in a transformation must be the diagonals of a convex quadrilateral (in the present triangulation), a parallel algorithm solves the problem in $O(n)$ independent replacements. Yet, for the general case of edge replacements and for all types of triangulations, $O(n)$ processors operating in parallel can solve the problem *in constant time*, where each processor replaces one edge by another, leading to a superlinear speedup. This observation, reminiscent of the furniture-moving paradigm [5], motivated me to search for other instances of this phenomenon. A result beyond superlinear speedup is provided by convex decompositions (*CD*s) of point sets, that is, planar graphs each of whose faces is a convex polygon. Consider the *CD*s of $3n$ points A and B in Fig. 2. Here *no sequential algorithm* can transform A into B by a sequence of transformations, each involving a constant number of edge replacements, such that every intermediate graph is a *CD*: any transformation involving fewer than n edge replacements creates a concavity. A parallel algorithm, however, easily performs the transformation in constant time [5].

References

[1] Akl, S.G., *Parallel Computation: Models And Methods*, Prentice Hall, 1997.

[2] Akl, S.G., Computing in the presence of uncertainty: Disturbing the peace, *Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications*, Las Vegas, Nevada, June 2003, Vol. I, pp. 442 - 448.

[3] Akl, S.G., Superlinear performance in real-time parallel computation, *The Journal of Supercomputing*, Vol. 29, No. 1, 2004, pp. 89 - 111.

[4] Akl, S.G., Inherently parallel geometric problems, Technical Report No. 2004-480, School of Computing, Queen's University, Kingston, Ontario, April 2004, 19 pages.

[5] Akl, S.G., Secure file transfer: A computational analog to the furniture moving paradigm, *Parallel and Distributed Computing Practices*, Vol. 5, No. 2, 2004, pp. 193 - 203.

[6] Akl, S.G., Cordy, B., and Yao, W., An analysis of the effect of parallelism in the control of dynamical systems, to appear in the *International Journal of Parallel, Emergent and Distributed Systems.*

[7] Krishna, C.M. and Shin, K.G., *Real-Time Systems*, Mc-Graw Hill, 1997.