A PRAM-on-chip Vision
(Invited Abstract)

Uzi Vishkin
University of Maryland Institute for Advanced Computer Studies (UMIACS), and Electrical and Computer Engineering
vishkin@umiacs.umd.edu

Explicit Multi-Threading (XMT) is a fine-grained computation framework introduced in our SPAA’98 paper. XMT aims at faster single-task completion time by way of instruction-level parallelism (ILP). Building on some key ideas from parallel computing, XMT covers the spectrum from algorithms through architecture to implementation; the main implementation related innovation in XMT was through the incorporation of low-overhead hardware mechanisms (for more effective fine-grained parallelism).

If and when implemented, XMT will enable the algorithm designer and the programmer to reason about parallel algorithms in a very simple parallel computation model, known as the PRAM (for parallel random-access-machine). The algorithms/theory community has developed PRAM algorithmics during the past two decades to the level where it appears to be second in magnitude only to serial algorithmics. Many elegant techniques and paradigms have been introduced. However, the evolution of multiprocessors never reached a situation where the PRAM model offered effective abstraction for them. So, this elegant theory has not been matched by an experimental study of what works better, more refined performance measurements, and a broad study of applications. In particular, the general question “how good parallel algorithms can really be” remained generally open.

A year 2000 perspective is that: Through XMT, the upcoming “on-chip Billion transistor” era calls for a massive revisit of PRAM-related algorithmics. “PRAM-on-chip” is an original direction for addressing the fascinating question: What to do with all the on-chip hardware once the returns on adding more on-chip memory start to diminish? This direction is the main innovation of XMT.

A few pointed examples of PRAM revisits will be provided. Although a good first step, there is much more to do beyond just examining previous approaches and get the most practical algorithm out of them. The examples will demonstrate:

(i) Understanding the profound differences between XMT and more traditional parallel computing implementation platforms/models.
(ii) Seeking good speedups for bounded size inputs.
(iii) Incorporating rigorous non-asymptotic performance analysis into experimental performance analysis.
(iv) Providing examples where experimental research and theoretical analysis complement one another.

Work in progress includes, studying major application domains, compiler and system-related work, as well as VLSI-related work. Overall, XMT provides ample research opportunities through mixing theory and experimentation.

The XMT home page is at: http://www.umiacs.umd.edu/vishkin/XMT