6.14 Concluding remarks

(Original section[1])

> "We are dedicating all of our future product development to multicore designs. We believe this is a key in inflection point for the industry. … This is not a race. This is a sea change in computing …"
> *Paul Otellini, Intel President, Intel Developers Forum, 2004*

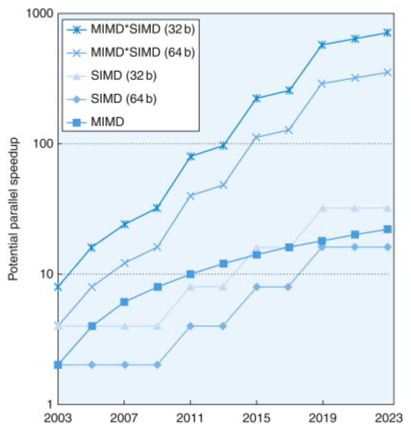
The dream of building computers by simply aggregating processors has been around since the earliest days of computing. Progress in building and using effective and efficient parallel processors, however, has been slow. This rate of progress has been limited by difficult software problems as well as by a long process of evolving the architecture of multiprocessors to enhance usability and improve efficiency. We have discussed many of the software challenges in this chapter, including the difficulty of writing programs that obtain good speed-up due to Amdahl's Law. The wide variety of different architectural approaches and the limited success and short life of many of the parallel architectures of the past have compounded the software difficulties. We discuss the history of the development of these multiprocessors in online COD Section 6.15 (Historical perspective and further reading). To go into even greater depth on topics in this chapter, see Chapter 4 (Data-Level Parallelism in Vector, SIMD, and GPU Architectures) of *Computer Architecture: A Quantitative Approach, Fifth Edition* for more on GPUs and comparisons between GPUs and CPUs and Chapter 6 (Parallel Processor from Client to Cloud) for more on WSCs.

As we said in COD Chapter 1 (Computer Abstractions and Technology), despite this long and checkered past, the information technology industry has now tied its future to parallel computing. Although it is easy to make the case that this effort will fail like many in the past, there are reasons to be hopeful:

- Clearly, *software as a service* (SaaS) is growing in importance, and clusters have proven to be a very successful way to deliver such services. By providing redundancy at a higher level, including geographically distributed datacenters, such services have delivered 24 × 7 × 365 availability for customers around the world.
- We believe that Warehouse-Scale Computers are changing the goals and principles of server design, just as the needs of mobile clients are changing the goals and principles of microprocessor design. Both are revolutionizing the software industry as well. Performance per dollar and performance per joule drive both mobile client hardware and the WSC hardware, and parallelism is the key to delivering on those sets of goals.
- SIMD and vector operations are a good match to multimedia applications, which are playing a larger role in the Post-PC Era. They share the advantage of being easier for the programmer than classic parallel MIMD programming and being more energy-efficient than MIMD. To put into perspective the importance of SIMD versus MIMD, the figure below plots the number of cores for MIMD versus the number of 32-bit and 64-bit operations per clock cycle in SIMD mode for x86 computers over time. For x86 computers, we expect to see two additional cores per chip about every 2 years and the SIMD width to double about every 4 years. Given these assumptions, over the next decade the potential speed-up from SIMD parallelism is twice that of MIMD parallelism. Given the effectiveness of SIMD for multimedia and its increasing importance in the post-PC Era, that emphasis may be appropriate. Hence, it's as least as important to understand SIMD parallelism as MIMD parallelism, even though the latter has received much more attention.

Figure 6.14.1: Potential speed-up via parallelism from MIMD, SIMD, and both MIMD and SIMD over time for x86 computers (COD Figure 6.28).

This figure assumes that two cores per chip for MIMD will be added every 2 years and the number of operations for SIMD will double every 4 years.



- The use of parallel processing in domains such as scientific and engineering computation is popular. This application domain has an almost limitless thirst for more computation. It also has many applications that have lots of natural concurrency. Once again, clusters dominate this application area. For example, using the 2012 Top 500 report, clusters are responsible for more than 80% of the 500 fastest Linpack results.
- All desktop and server microprocessor manufacturers are building multiprocessors to achieve higher performance, so, unlike in the past, there is no easy path to higher performance for sequential applications. As we said earlier, sequential programs are now slow programs. Hence, programmers who need higher performance *must* parallelize their codes or write new parallel processing programs.
- In the past, microprocessors and multiprocessors were subject to different definitions of success. When scaling uniprocessor performance, microprocessor architects were happy if single thread performance went up by the square root of the increased silicon area. Thus, they were pleased with sublinear performance in terms of resources. Multiprocessor success used to be defined as *linear* speed-up as a function of the number of processors, assuming that the cost of purchase or cost of administration of *n* processors was *n* times as much as one processor. Now that parallelism is happening on-chip via multicore, we can use the traditional microprocessor metric of being successful with sublinear performance improvement.
- The success of just-in-time runtime compilation and autotuning makes it feasible to think of software adapting itself to take advantage of the increasing number of cores per chip, which provides flexibility that is not available when limited to static compilers.
- Unlike in the past, the open source movement has become a critical portion of the software industry. This movement is a meritocracy, where better engineering solutions can win the mind share of the developers over legacy concerns. It also embraces innovation,

inviting change to old software and welcoming new languages and software products. Such an open culture could be extremely helpful during this time of rapid change.

To motivate readers to embrace this revolution, we demonstrated the potential of parallelism concretely for matrix multiply on the Intel Core i7 (Sandy Bridge) in the Going Faster sections of COD Chapters 3 (Arithmetic for Computers) to 6 (Parallel Processor from Client to Cloud):

- Data-level parallelism in COD Chapter 3 (Arithmetic for Computers) improved performance by a factor of 3.85 by executing four 64-bit floating-point operations in parallel using the 256-bit operands of the AVX instructions, demonstrating the value of SIMD.
- Instruction-level parallelism in COD Chapter 4 (The Processor) pushed performance up by another factor of 2.3 by unrolling loops four times to give the out-of-order execution hardware more instructions to schedule.
- Cache optimizations in COD Chapter 5 (Large and Fast: Exploiting Memory Hierarchy) improved performance of matrices that didn't fit into the L1 data cache by another factor of 2.0 to 2.5 by using cache blocking to reduce cache misses.
- Thread-level parallelism in this chapter improved performance of matrices that don't fit into a single L1 data cache by another factor of 4 to 14 by utilizing all 16 cores of our multicore chips, demonstrating the value of MIMD. We did this by adding a single line using an OpenMP pragma.

Using the ideas in this book and tailoring the software to this computer added 24 lines of code to DGEMM. For the matrix sizes of 32×32, 160×160, 480×480, and 960×960, the overall performance speed-up from these ideas realized in those two-dozen lines of code is factors of 8, 39, 129, and 212!

This parallel revolution in the hardware/software interface is perhaps the greatest challenge facing the field in the last 60 years. You can also think of it as an outstanding opportunity, as our Going Faster sections demonstrate. This revolution will provide many new research and business prospects inside and outside the IT field, and the companies that dominate the multicore era may not be the same ones that dominated the uniprocessor era. After understanding the underlying hardware trends and learning to adapt software to them, perhaps you will be one of the innovators who will seize the opportunities that are certain to appear in the uncertain times ahead. We look forward to benefiting from your inventions!

(*1) This section is in original form.