APPENDIX

Further Reading

A

We have carefully avoided diving into topics, no matter how interesting, that were not critical to teaching the keys to structured (parallel) programming. In this appendix, we offer some recommendations on where to learn more about closely associated topics. A deeper understanding of any of these will strengthen your understanding of what is behind good structured programming.

A.1 PARALLEL ALGORITHMS AND PATTERNS

Much has been written on parallel algorithms. We recommend reading a landmark paper focused on classifying algorithms, "The Landscape of Parallel Computing Research: A View from Berkeley" [ABC+06]. This paper used **dwarfs** as the name for common recurring **patterns** found in applications. Later papers switched to calling these **motifs**. This is an excellent starting point, but we also recommend the OUR pattern language [Par11]. This web site contains a set of collaboratively created pattern definitions.

"Introduction to Algorithms" [CLRS09] is a good general introduction to algorithms which also makes some usage of Cilk Plus.

A thorough and scholarly book specifically on parallel patterns is *Patterns for Parallel Programming* [MSM04]. Even though we do not recommend it specifically for parallel programming, we will note that the landmark book to promote patterns for software development is *Design Patterns: Elements of Reusable Object-Oriented Software* [GHJV95]. It is often affectionately called the "gang of four" book.

The key conferences involved in parallel algorithm development would be Symposium on Principles and Practice of Parallel Programming (PPoPP), ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), International Supercomputing in Europe (ISC), and Supercomputing in North America (SC). PPoPP, ISC, and SC are known for notable training opportunities via tutorials held in conjunction with their respective general conferences.

A.2 COMPUTER ARCHITECTURE INCLUDING PARALLEL SYSTEMS

The gold standard for teaching computer architecture is *Computer Architecture: A Quantitative Approach* [HP07] by John L. Hennessy and David A. Patterson. A new book, often described as very approachable and which caters to programmers, is *Computer Systems: A Programmer's*

Perspective [REB11] by Randal Bryant and Dave O'Hallaron. Both of these books explain the fundamental concepts in computer architecture.

For those whose interest runs deep enough to want to engage others with a keen interest in computer architecture, four long-standing conferences are particularly worth investigating: International Symposium on Computer Architecture (ISCA), Architectural Support for Programming Languages and Operating Systems (ASPLOS), International Symposium on High Performance Computer Architecture (HPCA), and International Conference on Parallel Architectures and Compilation Techniques (PACT).

A.3 PARALLEL PROGRAMMING MODELS

Intel Threading Building Blocks [Rei07] is a book that introduces TBB and provides many examples. The book predates **lambda expressions** and a number of additional advanced features of TBB. The documentation that accompanies TBB (`http://threadingbuildingblocks.org`) is exceptional and a reliable source of current information on all the features of the latest TBB.

The original Cilk home at the Massachusetts Institute of Technology (MIT) continues to maintain useful samples, tutorials, and papers about Cilk. Visiting it at `http://supertech.csail.mit.edu/cilk/` is recommended to learn more about Cilk. Intel and the related open source project for Cilk Plus maintain the web site `http://cilkplus.org`.

Array Building Blocks, likewise, has a set of examples, tutorials, and downloads hosted by Intel at `http://intel.com/go/arbb`.

The gold standard for learning MPI is *Parallel Programming with MPI* [Pac96]. A good book for learning more about both Open MP and MPI is *Parallel Programming in C with MPI and OpenMP* [Qui03].

If you have an interest in parallel programming for Java, we recommend reading *Java Concurrency in Practice* [PGB+05]. Doug Lea, in particular, has published books on parallel programming for Java that have been exceptionally useful for learning parallel programming concepts regardless of your programming language of choice.

If you want to get a handle on a number of other approaches in programming languages for parallelism, we recommend *Seven Languages in Seven Weeks: A Pragmatic Guide to Learning Programming Languages* [Tat10] by Bruce A. Tate, which examines Clojure, Haskell, Io, Prolog, Scala, Erlang, and Ruby. It is not likely that you will use all seven, but you will gain an appreciation for the many possibilities in designing support for concurrency into programming languages.

Finally, *The Art of Multiprocessor Programming* [HS08] covers a broad set of concerns for parallel programming in an approachable manner with a gentle mix of fables and stories.

The key conferences to consider attending include PPoPP, ISC, and SC, which are spelled out in Section A.1. All three are known for their tutorials, which are usually held just before the general conferences.

We will also recommend Herb Sutter's paper, "The Free Lunch Is Over: A Fundamental Turn Towards Concurrency in Software" [Sut05] which is often cited for how effectively it called attention to the shift to multicore parallelism and the challenge it represents.

Finally, if you have any doubts about the need to abandon **threads** in programming in favor of **tasks**, "The Problem with Threads" [Lee06] is recommended reading. We like to think of it as a modern version of the classic "Go To Statement Considered Harmful" [Dij68], which has come to be so commonly accepted today that it is hard to recall the controversy it raised for more than a decade after its publication. We are confident that programming with threads will go the way of Go To, and these are the two key papers that are heralding the change.