2. *Learning with Kernels (Support Vector Machines, Regularization, Optimization, and Beyond)* Bernard Scholkopf and Alexander Smola.

3. *Learning Kernel Classifiers* by Herbrich.

4. *Boolean Functions and Computation Models* by Clote and Kranakis.

Review of **Algorithms Sequential & Parallel: A Unified Approach** [2]
Authors of Book: R. Miller & L. Boxer
Prentice Hall 2000, 330 pages, $68.00
Reviewed by Anthony Widjaja <twidjaja@acm.org> Melbourne University

1 Introduction

Nowadays, the study of parallel algorithms has become a more mainstream topic in computer science partly because of the difficult problems in fields such as computational biology and computational geometry, to name a few. There have been several different approaches to teaching this topic at university level with which computer science educators around the world have come up. The authors of this book, Miller & Boxer, took a novel approach to the design and analysis of algorithms by unifying those of sequential and parallel algorithms. More importantly, the authors suggest that several courses based on this book has been taught successfully at both undergraduate and graduate levels at the State University of New York at Buffalo.

Prerequisites *(suggested by the authors)*:

- basic knowledge on data structures (e.g. stacks, queues, lists, trees) with which the reader who has taken a CS2 course must be familiar. This would imply the need for knowledge on fundamental algorithms such as sorting and searching. This subject is covered thoroughly in [1] and [2].
- fundamental discrete maths and calculus, especially summations, integrals, and limits.

In short, the readers, starting from junior college, will find this book accessible.

2 Summary of Contents

Chapters 1,2 & 3 contain the mathematical preliminaries required in later chapters. In particular, Miller & Boxer review some fundamentals of asymptotic analysis in chapter 1. Thereafter, in chapter 2, the authors revisit concepts of mathematical induction and recursion. As an aside, I was quite surprised to see the proof of the principle of mathematical induction in this chapter as I had initially thought that it was an axiom, which we cannot prove. The proof uses the **greatest lower bound axiom** (some people call it the "least number principle"), which essentially says that any subset of natural numbers has a greatest

[2]

lower bound. In chapter 3, the master method, a cookbook-type of system for evaluating recurrence relations, is presented. The proof of the master theorem, upon which the master method is based, is also presented in gory detail.

Chapter 4 concentrates on *combinatorial circuits* and *sorting networks*. This chapter is used as a motivation for the study of parallel algorithms in later chapters. In particular, the *bitonic sort*, which is an example of parallel sorting algorithms, is presented in fair amount of detail. In chapter 5, Miller & Boxer introduce basic models of sequential and parallel computation. Initially, the authors introduce the RAM (Random Access Machine) model as a model of sequential computation, and the PRAM (Parallel Random Access Machine) model as a model of parallel computation. Later in the chapter some other parallel models of computation including the mesh, tree, pyramid, mesh-of-trees, and hypercube are discussed in a thorough manner.

One ubiquitous example of parallel algorithms is the operation of matrix multiplication. In chapter 6, the authors begin to draw the reader's attention to some aspects of the field of scientific computing, in which many complex problems reside. In particular, the reader will see sequential and parallel algorithms for carrying out matrix multiplication and Gaussian elimination.

In chapter 7, the reader will learn the concept of *parallel prefix*, a powerful operation (especially in parallel computers), and efficient algorithms for performing this operation. Some basic applications of this operation are stipulated in this chapter, while some more advanced ones in later chapters.

Chapter 8 presents an implementation of linked lists for parallel architecture, the PRAM in particular. An interesting feature of this implementation is that it allows the computer to perform a *pointer-jumping algorithm*, which can speed up a linked-list traversal from the head to the last item of the list up to O(log n). After that, the linked-list implementation of parallel prefix, whose array implementation was introduced in the previous chapter, is presented.

In chapter 9, the reader will explore more into the design and analysis of divide-and-conquer algorithms using both sequential and parallel approaches. Some algorithms the reader will find are including mergeSort, quickSort and hyperQuickSort (a parallel implementation of quickSort).

Chapter 10, 11, 12, & 13 are a selection of (more advanced) topics in which the techniques introduced in previous chapters are extensively used. In specific, chapter 10 concerns with problems and solutions in the field of computational geometry. In chapter 11, Miller & Boxer delve into the field of image processing. Chapter 12 & 13 cover, respectively, topics of graph algorithms and numerical problems. All these topics are presented in quite a detailed fashion.

3 Opinions

First of all, I'm happy to report that the "story" told by this book is quite easy to follow. In particular, I like the way the authors present the material. That is, motivations are always given when introducing new concepts and also that comparisons of the running time complexity of algorithms in different models of computation are given. In addition,

this book is quite self-contained in the sense that every theorem presented in this book is proved. Also, a fair number of examples are worked out in great detail using a variety of methods. That is, for example, given a problem, the authors first present a solution with a sequential algorithm and, in turn, introduce the parallel counterpart which often has lower complexity. Furthermore, there are a moderate number of exercises given at the end of each chapter. Unfortunately, the readers who wish to consult an "exact" answer to each question in the exercises will be disappointed. However, almost every question that I considered hard has some hints, which helped me solve them in the end. Finally, it is important to note that the second printing of the book (which I have) has undergone quite a big change because of the relatively large number of small (mostly typographyical) errors found in the first printing (mentioned by one of the authors in his website). The website which contains errata, corrections, and supplementary materials for this book is available at `<http://www.prenhall.com/millerbox>` and updated quite regularly, which shows the author's dedication for the quality of the book.

This book can serve as a prescribed textbook for an advanced algorithms course, which focuses on the design and analysis of sequential and parallel algorithms. If the reader intends to use the book for this purpose, some suggestions as to the structure of the course can be found on the companion website. The reader will also find a set of lecture notes there. Also, I think this book is suitable to be read cover-to-cover since almost always the concepts in subsequent chapters are based on those of the previous ones. Some chapters, though, I think are more important than others. For example, chapter 4 to 9 cover the essence of the design and analysis of sequential & parallel algorithms, while chapter 10 to 13 provide some applications of them.

However, this book is not about parallel programming such as "cluster computing in Linux". In fact, the authors emphasized that most of the parallel models of computation introduced in this book are not yet physically realizable due to current technological limitations in connecting processors and memory (see page 82). That is, RAM and PRAM models assume "unbounded parallelism", in the sense that we can hook up as many processors as we like (but we want to minimize them, of course)[3]. Furthermore, the authors use pseudocodes when presenting algorithms, which make things slightly worse for those only interested in the practical side of parallel algorithms. Nevertheless, this book should please everyone with a taste for theoretical computer science, who is presumably new to parallel algorithms and is curious as to what advantages parallel model of computation offers to us in comparison to its sequential counterpart.

All in all, I will give five stars (out of five) for this book.

References

[1] R. Sedgewick. *Algorithms in C.* Addison Wesley, 1998.

[2] T. Cormen, et al. *Introduction to Algorithms*, second edition. MIT Press, 2001.

[3]Notice that if fully unbounded parallelism is physically realizable, we can solve NP-complete problems in polynomial time