Preface

Most of the professional programmers that I've encountered are not well prepared to tackle algorithm design problems. This is a pity, because the techniques of algorithm design form one of the core practical *technologies* of computer science. Designing correct, efficient, and implementable algorithms for real-world problems is a tricky business, because the successful algorithm designer needs access to two distinct bodies of knowledge:

- *Techniques* – Good algorithm designers understand several fundamental algorithm design techniques, including data structures, dynamic programming, depth-first search, backtracking, and heuristics. Perhaps the single most important design technique is *modeling*, the art of abstracting a messy real-world application into a clean problem suitable for algorithmic attack.

- *Resources* – Good algorithm designers stand on the shoulders of giants. Rather than laboring from scratch to produce a new algorithm for every task, they know how to find out what is known about a particular problem. Rather than reimplementing popular algorithms from scratch, they know where to seek existing implementations to serve as a starting point. They are familiar with a large set of basic algorithmic problems, which provides sufficient source material to model most any application.

This book is intended as a manual on algorithm design, providing access to both aspects of combinatorial algorithms technology for computer professionals and students. Thus this book looks considerably different from other books on algorithms. Why?

- We reduce the design process to a sequence of questions to ask about the problem at hand. This provides a concrete path to take the nonexpert from an initial problem statement to a reasonable solution.

- Since the practical person is usually looking for a program more than an algorithm, we provide pointers to solid implementations whenever they are available. We have collected these implementations on the enclosed CD-ROM and at one central WWW site for easy retrieval. Further, we provide recommendations to make it easier to identify the correct code for the job. With these implementations available, the critical issue in algorithm design becomes properly modeling your application, more so than becoming intimate with the details of the actual algorithm. This focus permeates the entire book.

- Since finding out what is known about a problem can be a difficult task, we provide a catalog of important algorithmic problems as a major component of this book. By browsing through this catalog, the reader can quickly identify what their problem is called, what is known about it, and how they should proceed to solve it. To aid in problem identification, we include a pair of "before" and "after" pictures for each problem, illustrating the required input and output specifications.

- For each problem in the catalog, we provide an honest and convincing motivation, showing how it arises in practice. If we could not find such an application, then the problem doesn't appear in this book.

- In practice, algorithm problems do not arise at the beginning of a large project. Rather, they typically arise as subproblems when it suddenly becomes clear that the programmer does not know how to proceed or that the current program is inadequate. To provide a better perspective on how algorithm problems arise in the real world, we include a collection of "war stories," tales from our experience on real problems. The moral of these stories is that algorithm design and analysis is not just theory, but an important tool to be pulled out and used as needed.

Equally important is what we do not do in this book. We do not stress the mathematical analysis of algorithms, leaving most of the analysis as informal arguments. You will not find a single theorem anywhere in this book. Further, we do not try to be encyclopedic in our descriptions of algorithms, but only in our pointers to descriptions of algorithms. When more details are needed, the reader should follow the given references or study the cited programs. The goal of this manual is to get you going in the right direction as quickly as possible.

But what is a manual without software? This book comes with a substantial electronic supplement, an ISO-9660 compatible, multiplatform CD-ROM, which can be viewed using Netscape, Microsoft Explorer, or any other WWW browser. This CD-ROM contains:

- A complete hypertext version of the full printed book. Indeed, the extensive cross-references within the book are best followed using the hypertext version.

- The source code and URLs for all cited implementations, mirroring the Stony Brook Algorithm Repository WWW site. Programs in C, C++, Fortran, and Pascal are included, providing an average of four different implementations for each algorithmic problem. Over 50 megabytes of software is included.

- More than thirty hours of *audio* lectures on the design and analysis of algorithms are provided, all keyed to the on-line lecture notes. Following these lectures provides another approach to learning algorithm design techniques. Listening to all the audio is equivalent to taking a one-semester college course on algorithms!

This book is divided into two parts, *techniques* and *resources*. The former is a general guide to techniques for the design and analysis of computer algorithms. The resources section is intended for browsing and reference, and comprises the catalog of algorithmic resources, implementations, and an extensive bibliography.

Altogether, this book covers material sufficient for a standard *Introduction to Algorithms* course, albeit one stressing design over analysis. We assume the reader has completed the equivalent of a second programming course, typically titled *Data Structures* or *Computer Science II*. Textbook-oriented features include:

- In addition to standard pen-and-paper exercises, this book includes "implementation challenges" suitable for teams or individual students. These projects and the applied focus of the text can be used to provide a new laboratory focus to the traditional algorithms course. More difficult exercises are marked by (*) or (**).

- "Take-home lessons" at the beginning of each chapter emphasize the concepts to be gained from the chapter.

- This book stresses design over analysis. It is suitable for both traditional lecture courses and the new "active learning" method, where the professor does not lecture but instead guides student groups to solve real problems. The "war stories" provide an appropriate introduction to the active learning method.

- A full set of lecture slides for teaching this course is available on the CD-ROM and via the World Wide Web, both keyed to unique on-line audio lectures covering a full-semester algorithm course. Further, a complete set of my videotaped lectures using these slides is available for interested parties. See http://www.cs.sunysb.edu/~algorith for details.

Acknowledgments

I would like to thank several people for their concrete contributions to this project. Ricky Bradley built up the substantial infrastructure required for both the WWW site and CD-ROM in a logical and extensible manner. Zhong Li did a spectacular job drawing most of the catalog figures using xfig and entering the lecture notes that served as the foundation of Part I of this book. Frank Ruscica, Kenneth McNicholas and Dario Vlah came up big in the pinch, redoing all the audio and helping as the completion deadline approached. Filip Bujanic, David Ecker, David Gerstl, Jim Klosowski, Ted Lin, Kostis Sagonas, Kirsten Starcher, Brian Tria, and Lei Zhao all made contributions at various stages of the project.

Richard Crandall, Ron Danielson, Takis Metaxas, Dave Miller, Giri Narasimhan, and Joe Zachary all reviewed preliminary versions of the manuscript and/or CD-ROM; their thoughtful feedback helped to shape what you see here. Thanks also to Allan Wylde, the editor of my previous book as well as this one, and Keisha Sherbecoe and Robert Wexler of Springer-Verlag.

I learned much of what I know about algorithms along with my graduate students Yaw-Ling Lin, Sundaram Gopalakrishnan, Ting Chen, Francine Evans, Harald Rau, Ricky Bradley, and Dimitris Margaritis. They are the real heroes of many of the war stories related within. Much of the rest I have learned with my Stony Brook friends and colleagues Estie Arkin and Joe Mitchell, who have always been a pleasure to work and be with.

Finally, I'd like to send personal thanks to several people. Mom, Dad, Len, and Rob all provided moral support. Michael Brochstein took charge of organizing my social life, thus freeing time for me to actually write the book. Through his good offices I met Renee. Her love and patience since then have made it all worthwhile.