SOFTWARE FOUNDATIONS

VOLUME 3: VERIFIED FUNCTIONAL ALGORITHMS

TABLE OF CONTENTS INDEX ROADMAP

PREFACE

Welcome

Here's a good way to build formally verified correct software:

- Write your program in an expressive language with a good proof theory (the Gallina language embedded in Coq's logic).
- Prove it correct in Coq.
- Compile it with an optimizing ML compiler.

Since you want your programs to be *efficient*, you'll want to implement sophisticated data structures and algorithms. Since Gallina is a *purely functional* language, it helps to have purely functional algorithms.

In this volume you will learn how to specify and verify (prove the correctness of) sorting algorithms, binary search trees, balanced binary search trees, and priority queues. Before using this book, you should have some understanding of these algorithms and data structures, available in any standard undergraduate algorithms textbook.

This electronic book is Volume 3 of the *Software Foundations* series, which presents the mathematical underpinnings of reliable software. It builds on *Software Foundations Volume 1* (Logical Foundations), but does not depend on Volume 2. The exposition here is intended for a broad range of readers, from advanced undergraduates to PhD students and researchers.

The principal novelty of *Software Foundations* is that it is one hundred percent formalized and machine-checked: the entire text is literally a script for Coq. It is intended to be read alongside an interactive session with Coq. All the details in the text are fully formalized in Coq, and the exercises are designed to be worked using Coq.

Practicalities

Chapter Dependencies

Before using *Verified Functional Algorithms*, read (and do the exercises in) these chapters of *Software Foundations Volume I*: Preface, Basics, Induction, Lists, Poly, Tactics, Logic, IndProp, Maps, and perhaps (ProofObjects), (IndPrinciples).

In this volume, the core path is:

Preface -> Perm -> Sort -> SearchTree -> Redblack

with many optional chapters whose dependencies are,

- Sort -> Multiset or Selection or Decide
- SearchTree -> ADT or Extract
- Perm -> Trie
- Sort -> Selection -> SearchTree -> ADT -> Priqueue -> Binom

The Color chapter is advanced material that should not be attempted until the student has had experience with most of the earlier chapters, or other experience using Coq.

System Requirements

Coq runs on Windows, Linux, and OS X. The Preface of Volume 1 describes the Coq installation you will need. This edition was built with Coq 8.7.1.

In addition, two of the chapters ask you to compile and run an OCaml program; having OCaml installed on your computer is helpful, but not essential.

Exercises

Each chapter includes numerous exercises. Each is marked with a "star rating," which can be interpreted as follows:

- One star: easy exercises that underscore points in the text and that, for most readers, should take only a minute or two. Get in the habit of working these as you reach them.
- Two stars: straightforward exercises (five or ten minutes).
- Three stars: exercises requiring a bit of thought (ten minutes to half an hour).
- Four and five stars: more difficult exercises (half an hour and up).

Also, some exercises are marked "advanced", and some are marked "optional." Doing just the non-optional, non-advanced exercises should provide good coverage of the core material. Optional exercises provide a bit of extra practice with key concepts and introduce secondary themes that may be of interest to some readers. Advanced exercises are for readers who want an extra challenge (and, in return, a deeper contact with the material).

Please do not post solutions to the exercises in any public place: Software Foundations is widely used both for self-study and for university courses. Having solutions easily available makes it much less useful for courses, which typically have graded homework assignments. The authors especially request that readers not post solutions to the exercises anyplace where they can be found by search engines.

Downloading the Coq Files

A tar file containing the full sources for the "release version" of this book (as a collection of Coq scripts and HTML files) is available at http://www.cis.upenn.edu/~bcpierce/sf.

(If you are using the book as part of a class, your professor may give you access to a locally modified version of the files, which you should use instead of the release version.)

Lecture Videos

Lectures on for an intensive summer course based on some chapters of this book at the DeepSpec summer school in 2017 can be found at https://deepspec.org/event/dsss17/lecture_appel.html.

For Instructors and Contributors

If you plan to use these materials in your own course, you will undoubtedly find things you'd like to change, improve, or add. Your contributions are welcome! Please see the Preface to *Logical Foundations* for instructions.

Thanks

Development of the *Software Foundations* series has been supported, in part, by the National Science Foundation under the NSF Expeditions grant 1521523, *The Science of Deep Specification*.