TP3
F593

AMAST Series in Computing: Vol. 7

Formal Models of Computation

The Ultimate Limits of Computing

Arthur Fleck

University of Iowa, USA

World Scientific
Singapore • New Jersey • London • Hong Kong

Published by

World Scientific Publishing Co. Pte. Ltd.

P O Box 128, Farrer Road, Singapore 912805

USA office: Suite 1B, 1060 Main Street, River Edge, NJ 07661

UK office: 57 Shelton Street, Covent Garden, London WC2H 9HE

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library.

FORMAL MODELS OF COMPUTATION
The Ultimate Limits of Computing

ISBN 981-02-4500-9

Printed in Singapore by Fulsland Offset Printing

Formal Models of Computation

The Ultimate Limits of Computing

Dedicated to Jean

for so many things, it would take another book to say.

Preface

Philosophical Orientation

The general goal of this book is to develop an understanding of the ultimate and insurmountable limits of computing. In order to understand these limits better, we seek to view them from both sides — to understand both what can be computed and what is inherently impossible. These limits vary depending on the amount and character of the available computing resources. The boundaries of what's possible are explored for a variety of combinations of computing resources. To this end, a varied collection of computational models is considered. These models are formulated at a high level of abstraction. The purpose of this abstraction is to ensure that the analysis applies to *any* suitable "computer". To achieve this level of abstraction, we develop precise (and machine independent) definitions, and follow them with rigorous deductions of the capabilities that are implied.

The material of this book introduces established formal models of computing. Developing the models involves understanding precise definitions, and rigorous reasoning based on these definitions. This precision and rigor provide a discipline that the author believes to be of great value to those working in computer science. On the other hand, formality cannot replace intuition. Insight and intuition are what provide the inspiration that guides technical development. However, intuition is by its very nature imprecise and therefore prone to error, and so an associated formalism is a necessity. A danger is that formality may obscure intuition. Formalism is developed in this book in the spirit of a ratification of the associated intuition, and the presentation weaves intuition and precision into a single fabric of insight and understanding. A significant strength of this book is that neither rigor nor intuition is sacrificed for the sake of the other. Rigor is developed as a refinement of intuition, and each is used to deepen the appreciation of the other.

This approach to understanding the inherent limitations of computation was initiated even before the invention of electronic computers! The profound work by Alan Turing provided an inspiration for the kind of deep insights that could be achieved through theoretical investigations. Turing was pursuing questions first raised in formal logic and his work pre-dated the development of general-purpose electronic computers. When John von Neumann. was later formulating his ideas for these devices, he did interact with Turing. Nevertheless, the initial stages of development of electronic digital computers proceeded on an ad hoc basis. The subsequent recognition of computer science as a distinct intellectual discipline was initially largely motivated by theoretical studies along the lines presented in this book.

One of the features of this book is that in addition to the most familiar models, we introduce a number of variations and applications along the way. The variations are not essential to the main thread of the development, and sections that elaborate them can safely be omitted without loss of continuity. However, the selection of a few that attract the reader's interest can augment the main line of investigation and help achieve a deeper understanding. By exhibiting alternative models and examining their properties, we seek to shed light on what may appear to be arbitrary choices in the standard models. In addition, the brief digressions into a few practical applications are intended to keep the reader's motivation at a high level — a crucial ingredient in the learning process.

Comments for Readers

Books that develop the basic theoretical ideas covered here usually fall into one of two categories. One approach is to begin with the most general models (e.g., Turing machines) and proceed through a series of restrictions on available resources and capabilities to simpler and simpler models, studying the reduction in the limits of computations that result. The second approach is the exact opposite, namely to begin with the simplest model (i.e., finite state machines) and consider a series of extensions of the computational resources and capabilities and the resulting expansions in computational power. The approach taken in this book is the latter. The finite state component persists throughout each of the enriched models of computing, and mastering the understanding of this aspect at the outset eases the way for the increasing complication. We take advantage of this in the presentation, and feel it aids in learning the material.

The material in this area is intellectually deep, and it is highly challenging to master. This book is unrelenting in its attention to the blending of intuition and rigor that is required to fully understand the area. This is one of the keys to achieving proficiency in this area. The intuitive perspective enters not only in the examples and discussions of the results, but also in the proofs. In fact, it is really in the formal proofs that intuition and rigor are most advantageously intertwined. The proofs are used not just to justify the results, but as a vehicle to create deeper insight.

This book emerged from notes and lectures resulting from many years of teaching the material. The level of the presentation assumes that the student is an upper-level undergraduate, or a beginning graduate student. The material presented covers all the topics suggested in the ACM curriculum guidelines for the Theory of Computation course[1,2]. The background assumed of the reader includes some general knowledge of computers and programming. We do not explicitly consider programs as such, although at a somewhat more abstract level the idea of algorithm is an emphasis of the book. However, for intuition and motivation, experience in programming computers is vital. The most crucial technical background is a good course in what is usually referred to in computer science curricula as "discrete structures". In particular, some prior exposure to abstraction, rigor, and methods of formal proof is assumed. Especially the idea of inductive proof is heavily used. A brief synopsis of the most critical supporting material is given in Chapter 0.

In the author's experience, working problems is an *essential* means of gaining a full understanding of the material. In this sense, there is a good analogy with learning computer programming — one can listen to lectures, read about programming, and study programs of others, but deeper understanding is not developed without actually writing programs yourself. So it is with the material in this book — to learn it well requires active participation, so work as many problems as you can. Therefore to this end, there are numerous problems at the end each chapter, and sample solutions

[1]A. B. Tucker (ed.), "Computing Curricula 1991, Report of the ACM/IEEE-CS Joint Curriculum Task Force", ACM order number 201910 (full report); also, *Commun. ACM* 34, 6 (June 1991), 68-84 (summary).

[2] H. M. Walker & G. M. Schneider, "A revised model curriculum for a liberal arts degree in computer science", *Commun. ACM* 39,12(Dec. 1996), 85-95.

for some are included in an appendix. The student is urged to attempt to solve a problem before consulting the sample solution. Also, note that these solutions are just samples and not the only correct ones, so if you devised a different solution, do not assume it is necessarily wrong or inferior.

Notational Conventions

Headings of optional sections are marked with the symbol ‡. We also use the special symbol ❑ to denote the ends of proofs and examples, and by this means seek to avoid any confusion by the reader about when a technical portion has been completed and where follow-up discussion about it begins. The problems with sample solutions are marked with ¤, and the most challenging problems are marked with the symbol *.

Electronic Links

The author's Web page (http://www.cs.uiowa.edu/~fleck/) contains a link to material related to the book, including an up-to-date errata. The author invites readers to email a report (fleck@cs.uiowa.edu) of any errors they detect.

Acknowledgements

I am appreciative of the reactions and questions of many past students for the improved presentation that has resulted. Also, my thanks go to Sungwon Kang and Hantao Zhang for their helpful comments on earlier versions of the manuscript. Finally, the enthusiastic support of Teodor Rus for getting this book into print is gratefully acknowledged.

Arthur C. Fleck
Iowa City, Iowa

Table of Contents

‡Section with this symbol can be omitted without loss of continuity.

Part III — General Computability Models

Chapter 0

Mathematical Preliminaries

Section 0.0: Introduction

While this book is about computing, it takes an abstract and formal approach in its investigations. At the heart of the content of this book is providing statements of the basic definitions with impeccable precision, and deducing the conclusions that flow from them with indisputable logic. We are mainly interested in establishing properties that entire categories of computations must possess rather than considering individual computations. This means that the ability to reason about formal systems is critical to following our development.

In this chapter we provide a succinct review of the mathematical prerequisites that the reader needs to understand in order to assimilate the ideas of this book. It is not intended that the coverage provided here will be sufficient for the reader with little or no prior exposure to this material, and this section is not a substitute for the study of appropriate background as outlined in the Preface. It is expected that the reader has previously studied these topics, and needs only to be reminded of some of these ideas. In fact, most readers should not need to do more than skim this chapter to align their previous background with the terminology and notation used in this book. Since terminology on these basic topics differs little from one writer to another, this should require minimal attention from readers with appropriate prerequisites.

Section 0.1: Sets and tuples

The most pivotal formalism used in this book to accomplish our goals is that of sets and their properties. The essential concept of *set* is both elementary and natural. It simply consists of grouping a number of distinct elements, the *members*, into a single unit, the set. Operations and properties that are common to all members of a set can then be treated once for the entire group rather than repeatedly for each of the members. Once we have some basic formalisms established, we will virtually always take the approach of building on them to develop the next one. However, as the first formalism to be introduced, sets cannot be explained in terms of a previously studied formalism. Therefore we must take another tack, normally either selecting abstract axioms that capture the relevant properties, or placing more reliance on informality and intuition. In this section we take the latter approach.

We will always employ sets within a specific *universe of discourse*. This gives us a definite context in which statements are to be interpreted. A statement such as "all persons over the age of 18 by election day are allowed to vote" is not sufficiently precise for our purposes. We need to know in addition, for example, that these persons are citizens of, for example, the United States. In such a case our universe of discourse might be all persons who are citizens of the United States (and we may wish to incorporate other constraints as well). A basic premise we will always assume is that there is a pre-determined collection — the **universe** — of all the objects that are in the domain of discussion. The universe will be selected as context for the issues we wish to discuss. For example in computing, one frequently used universe is the collection **Nat** of all natural numbers (i.e., non-negative integers), **Nat** = {0, 1, 2, 3, … }.

Once the universe is settled, to define a set we just need to describe how the elements of that set are determined. The key relationship for defining a set S is **membership** — if an object x from the universe is an element of S, we say that x **is a member of** S and write $x \in S$. There is no ordering between the members of a set that is implied by the fact of their membership. That is, sets are deliberately an unordered structure for aggregating elements. Also, elements never occur in a set repeatedly — either an element is a member of a set or it's not.

Sets may be defined in a number of ways. As long as it is clearly determined which elements are members of the set (and which are not), that is all that matters. One standard set notation is to write the elements of the set enclosed in "curly braces", sometimes called the *roster notation*. For instance, {2, 4, 6} denotes the set with the three members 2, 4, and 6 (say from the universe **Nat**). While we must pick an order to write down the elements of a set, this should not be interpreted as establishing an order between the elements. Thus {2, 4, 6} and {4, 2, 6} are two notations for the same set. Most of the sets of interest will be infinite, and for infinite sets we cannot simply list all the elements. The set notation is extended to permit the description of infinite sets with the following *set comprehension*, or *set abstraction* notation: {form of item | property of item}.

The "item form" portion of a set comprehension describes the form of potential members of the set, and the "property" portion describes a condition that must be satisfied in order for items of the indicated form to be members of the set — the set defined consists of *all* items of the indicated form having the specified property. For example, $\{n \in \textbf{Nat} \mid n>5\}$ describes the infinite set of all natural numbers greater than 5, {6, 7, 8, … }. Of course, with this notation the set that is described may be either finite or infinite. The last example is an infinite set, while $\{n \in \textbf{Nat} \mid 0 \leq n \leq 9\}$ is finite. Often the universe is implicitly clear from context and we may simply write $\{n \mid n>5\}$. Either the "item" description or the "property" description may be more involved. For instance, $\{2*n \mid n \in \textbf{Nat}\}$ describes the set of all even natural numbers.

Unrestricted use of informal definitions of sets can lead to paradoxes — logically impossible situations. No doubt the best known of these is Russell's paradox that is stated as follows: let r denote the set of all those sets which are not members of themselves, $\Re = \{S \text{ is a set} \mid S \notin S\}$. At first glance this appears to be a well-defined, but somewhat curious collection (since sets we normally conceive of are never members of themselves). However, if we ask the question of whether $\Re$ is a member of $\Re$ we find that this collection is much more than just "curious". Suppose that $\Re \in \Re$.