TLP **9** (2): 239–243, 2009.
doi:10.1017/S1471068409003731 Printed in the United Kingdom

Book review

Lambda-Calculus and Combinators, An Introduction, 2nd Edition, J. Roger Hindley and Jonathan P. Seldin, Cambridge University Press, 2008. Hardback, ISBN-13: 9780521898850, $70.00.

In short. This book is the second edition of the classical book *Introduction to Combinators and λ-calculus* by the same authors. The subject is λ-calculus and combinatory logic (CL), and the book gives a clear and well-written presentation of untyped λ-calculus and CL, typed versions of both systems, and their model theory.

What is new compared to the first edition? The title, a little bit. The much more modern typesetting. The order of what I call the three parts of the book: in the new version it is first the untyped systems, then the typed systems, and finally the model theory. In the old version it was untyped systems, model theory, and typed systems. An inspection of both tables of contents further reveals that most chapters have the same structure as in the first edition; in a few cases a section is added. Two chapters have been removed from the second edition: *Logic based on combinators* and *Gödel's consistency-proof for arithmetic*. The part concerned with typed systems has been rewritten considerably. Moreover, the formal details concerning renaming of bound variables have been moved from the introduction to λ-calculus to a separate new appendix. So the second edition differs quite significantly from the first.

Contents. The book consists of 16 chapters and 5 appendices. I see the chapters as divided into three parts. The first part, Chapters 1–9, is concerned with the syntax, expressivity, theories, and extensionality of untyped λ-calculus and CL. Then we switch to the typed setting in the second part, Chapters 10–13. This part treats simple typing à la Curry and à la Church quite extensively, and general typing systems more shortly. The third part, Chapters 14–16, turns back to the untyped world by presenting models for λ-calculus and CL.

Chapter 1 introduces untyped λ-terms, substitution, β-reduction, and β-equality. The section concerning substitution cites mainly the book by Curry and Feys from 1958 as source, and refers to Appendix A1 for details of proofs. The sections concerning β-reduction and β-equality treat the Church–Rosser property and confluence, and refer to Appendix A2 for proofs. The chapter contains some nice exercises. I liked in particular the exercises concerning normal forms and substitutions: find terms M and N such that $[N/x]M$ has a normal form but M does not, and, a harder one: find P and Q such that neither of P and Q has a normal form, but PQ has one. What I missed in this chapter is the representation of a λ-term as a tree; something I find often helpful for myself and in teaching.

Chapter 2 introduces untyped CL terms, weak reduction induced by the rules for the combinators **I**, **K**, and **S**, and weak equality. There are no issues concerning bound variables now. For proofs of confluence the reader is again referred to Appendix A2. Abstraction in CL is defined, but combinatory completeness is not yet shown.

Chapter 3 is concerned with both λ-calculus and CL, and starts by giving a fixed-point combinator for both systems. Here abstraction in CL, defined in Chapter 2, is used, and combinatorial completeness of CL is given as a corollary of the fixed-point theorem. The next topic concerning expressivity is Böhm's theorem, stating that syntactically different $\beta\eta$-normal forms can be distinguished by their behaviour. The statement makes use of the notion of η-reduction for λ-terms, and strong normal forms for CL. Both are defined in this chapter (one can imagine looking for them in Chapters 1 and 2). Then the quasi-leftmost reduction theorem is given, stating that by contracting repeatedly the leftmost redex the normal form of a term, if it exists, will be reached. The "quasi" means that such a rewrite sequence may even be "polluted" by steps that are not leftmost. This section contains a definition that I found rather puzzling: it states that a reduction *has maximal length* if it is either infinite or ends in a normal form. Now the reduction $(\lambda x.\, y)(\mathbf{I}\,\mathbf{I}) \to_\beta y$ is clearly not the longest possible; however, it has *maximal length* according to the definition. The word *length* in the definition is the thing that confuses me. The chapter concludes with an interesting and quite lengthy discussion, first on the history of λ-calculus and CL and their role in logic and computer science, and then more in particular on the interpretation of their (untyped) terms as operators.

Chapter 4 continues the study of expressivity by showing that all partial recursive functions can be represented in λ-calculus with β-equality and in CL with weak equality. The presentation is careful and detailed and hence very suitable for teaching. The chapter also shortly discusses representing arithmetic using an iteration operator.

Chapter 5 is a rather short chapter in which a general undecidability theorem is proved. This theorem states that no pair of non-empty sets of terms closed under conversion is recursively separable, both for the λ-calculus with β-conversion and for CL with weak conversion. Two important corollaries are: first, given a term there is no recursive way of deciding whether it has a normal form, and second, given two terms there is no recursive way of deciding whether they are convertible or not. These results, here stated as corollaries of a more general theorem, led Church to proving the undecidability of first-order predicate logic.

Chapter 6 is a proof-theoretic account of the formal theories of λ-calculus with β-equality and CL with weak equality. The word "formal" means that now these theories are presented using axioms, as for instance, $(\lambda x.\, M)\, N = [N/x]M$ and rules of inference. So now one can derive statements such as $\lambda\beta \vdash (\lambda x.\, x)\, y = y$, and one can talk about issues like admissible and derivable rules, and equivalence of theories. Equivalence of theories is used later in the book, for instance to talk about equivalent formalizations of extensionality. So I understand the use of the material presented in this chapter, but for me, it would not get top priority in teaching untyped λ-calculus.

Chapter 7 is concerned with extensionality in λ-calculus. It starts by presenting two rules and one axiom that have been proposed as a formalization of extensionality in

λ-calculus. The rule ζ roughly states that if M applied to x equals N applied to x, for a fresh variable x, then M equals N. The rule *ext* states that if $MP = NP$ for all P, then $M = N$. Finally, the axiom η states $\lambda x.\, M\, x = M$ with the side condition that x is not free in M. Now the terminology of Chapter 6 concerning rule equivalence of formal theories is used to state and prove that η is as strong as ζ and *ext*. The chapter then continues with a short study of λ-calculus with $\beta\eta$-reduction, showing confluence, hence uniqueness of normal forms, and η-postponement. The proofs of confluence are given in Appendix A2.

Chapter 8 treats the rather subtle subject of extensionality in CL. The theory of weak equality is extended to an extensional theory in two different ways: by adding an axiom ζ, similar to ζ for λ-calculus, and by adding ξ, also similar to the ξ-rule for λ-calculus, with the difference that abstraction in CL is no syntax but a meta-notation. One can instead add a rule η, similar to the η-axiom for λ-calculus, but this axiom holds in CL as an identity. Equivalence of adding ζ or adding ξ is proved, which is not hard. Then the issue whether there is a reduction theory for extensionality, like $\beta\eta$-reduction in λ-calculus, is tackled. This leads to the notion of strong reduction, for which confluence is proved, but, as the authors remark, it is a bit worrying that this proof uses the translation between λ-calculus and CL.

Chapter 9 is the last chapter of what I consider the first part of the book, concerned with untyped λ-calculus and CL. It treats the correspondence between λ-calculus and CL. This correspondence is stressed in the majority of the material treated so far, so the reader gets to know them as strongly related systems, but now the correspondence is studied in a formal way. Weak reduction, β-reduction, and different extensional equalities are discussed.

Now what I call part 2 of the book starts. It is concerned with typed λ-calculus and CL. This part contains the main differences compared to the first edition.

Chapter 10 introduces simple types and continues with defining simply typed λ-calculus à la Church. That means that we work with typed variables, and in an abstraction the type of the abstracted variable is part of the syntax: $\lambda x^{\sigma}.\, M$. Every subterm may be annotated with its type. One of the differences with the first edition of the book is that now simple types are written as $\sigma, \tau, \ldots$, which I think is much nicer than using $\alpha, \beta, \ldots$, because why use β for writing a type in a system with β-reduction?

Subject reduction and some elementary results concerning substitution are presented, before moving on to weak and strong normalization. For the proofs, the reader is referred to Appendix A3 where a proof using computability is presented.

Chapter 11 is concerned with simply typed CL à la Curry. The authors are experts on Curry typing, and indeed typing à la Curry for both λ-calculus and CL in Chapter 10 takes around 10 pages, whereas Chapters 11 and 12, dealing with typing à la Church for λ-calculus and CL, together take around 60 pages. So a wealth of material is presented. In Curry-style systems types are parametric, leading to a form of polymorphism à la ML. Then after some elementary results, subject reduction, and decidability of typing. The theorems of strong normalization of weak reduction and weak normalization of strong reduction are given. For the latter the reader is referred to the Italian edition of the predecessor of the first edition (where

also Lercher is a coauthor), together with a footnote concerning a typo. In a next incarnation of the book, it would be nice to give this proof explicitly. The chapter is rather long and contains more interesting issues: principal types, propositions as types, and equality.

Chapter 12, then, is concerned with simply typed λ-calculus à la Curry. The ideas and results are essentially equal to those of the previous chapter, concerning CL, but now there is the complication of bound variables. The complication is mainly in the introduction rule for $\rightarrow$, that is, the rule that tells us how to derive a term of type $\sigma \rightarrow \tau$. It corresponds to the rule for implication introduction in a Gentzen system for natural deduction. Abstraction of a variable in λ-calculus corresponds to discharging a hypothesis in the natural deduction proof system, and in both cases there is some administration to deal with. Some elementary properties leading to, for instance, subject reduction are discussed. Then the discussion moves on to principal types (the most general type that can be assigned to a term) and decidability of typability. The next topics are the correspondence between propositions and types, and weak and strong normalization. Again, the normalization proofs are mainly sketched, with a reference to the literature. Finally a system with equality is discussed.

Chapter 13 deals with more general typing systems. It starts with discussing a λ-calculus with dependent types à la Curry. It is very interesting to read because most presentations of dependent types use the Church-style of typing. Then the focus switches to Church-style typing indeed, to end eventually with a discussion of PTSs. One section is devoted to strong normalization of PTSs, presenting the skeleton of the proof due to Geuvers and Nederhof given in the chapter by Barendregt in the *Handbook of Logic in Computer Science*. While it is of course useful to mention the result, I doubt the use of presenting only the schedule of such a technical result; maybe here a reference to the literature would have been enough.

Now starts what I call part 3 of the book concerning model theory: Chapters 14, 15, and 16.

Chapter 14 is a nice and clear presentation of applicative structures and combinatory algebras as models of CL. It seems to me very suitable for teaching. On p. 227 an implication is given that is refuted by a counterexample due to Plotkin. The reader is referred to Remark 7.3 for this counterexample, but neither there his curiosity is satisfied: the counterexample is mentioned but unfortunately not explicitly given.

Chapter 15 treats the by nature more complex issues of models for the λ-calculus. Three kinds of models are presented: λ-models using an applicative structure, syntax-free λ-models, and Scott–Meyer λ-models.

Chapter 16 is more concrete and gives an extensive presentation of the model D_∞ due to Scott. Then some other models are discussed more shortly: the model D_A, the model P_ω, and filter models.

To conclude. To my taste, at some places the book could have been more explicit and at some other ones less explicit. Because personally I like examples very much. I was at a few places somewhat disappointed to see an example only mentioned without it being explicitly given. This happens for instance on p. 227 as mentioned

above, and then also for instance on pp. 230 and 235. So at such places I would have preferred a more explicit presentation. At some other places the book is more explicit than I think is needed. For instance in presenting the skeleton of the proof of strong normalization of PTSs, as mentioned above. Here for me a reference to the literature would have been enough, not so much because the proof can be found elsewhere but because with only the schedule of the proof I think the technical development is very hard to follow. In the new edition new references are added. I still missed one: a reference to *Term Rewriting Systems* by Terese, concerning the general theory of reduction. In Appendix A2 it could then be remarked that the confluence results also follow from a more general theorem concerning (weakly) orthogonal rewriting systems.

But the main point of my conclusion is that the book contains very interesting material, partly classical, and partly not easily found elsewhere. The presentation is very careful and clear. I appreciate very much the many exercises with solutions. The position of the book among other books on λ-calculi seems very good to me. The presentation is mathematically more advanced than for instance in the book *An Introduction to Lambda Calculi for Computer Scientists* by Hankin. On the other hand, it is shorter and more suitable for teaching than the encyclopedic book by Barendregt. Because of the focus on Curry-style typing systems it contains a lot of material which is not covered in Barendregt's chapter in the *Handbook of Logic in Computer Science*. So both for teaching and for research or self-study the book is an outstanding source with its own clear merits.

I think this second edition of this classical book is a beautiful asset for the literature on λ-calculus and CL.

Femke van Raamsdonk

https://doi.org/10.1017/S1471068409003731 Published online by Cambridge University Press