Review of
Communicating and Mobile Systems: The π-calculus[*]

Riccardo Pucella

Department of Computer Science
Cornell University

October 23, 2000

For a long time, the quest for a formal foundation of concurrent programming has kept semanticists happy and busy. What many were looking for was a calculus playing the same role for concurrent programming as the λ-calculus did for functional programming. One of the many difficulties encountered was that there are many aspects to concurrent programming and one has to decide what to consider primitive. In the early eighties, Milner introduced CCS [2] and Hoare introduced CSP [1] as calculi to model concurrent processes where communication (or interaction) between processes is taken as primitive. The main characteristic of these calculi was that one could reason algebraically about such processes, via equational laws from which one could define various notions of process equivalence. Two processes are deemed equivalent when they "have the same behavior" for some suitable notion of behavior.

In 1989, Milner extended CCS to take full advantage of named channels. To communicate over a channel, a process needs to know the name of that channel. Processes can moreover pass channel names over channels. The resulting calculus, the π-calculus, allows process A to pass the name of a channel c to another process B, at which point B can communicate via channel c to another process, with which B could not communicate before receiving the channel name. This feature enables the π-calculus to have a dynamic communication topology, which can be used to model mobility. In this setting, mobility is taken to be the mobility of links between components.

For the past ten years since the introduction of the π-calculus (ten years... recall that ten years ago, the first version of Microsoft Windows was barely out the door), the available literature on the π-calculus has consisted of original papers (for instance [5], [4], [7]) and PhD theses (for instance [6]). No longer.

Milner's short monograph, peaking at about 160 pages, gives a well-rounded and self-contained introduction to the material. According to the Preface, the book is based on lecture notes for a final-year undergraduate course, and the material is acknowledged to be challenging. No formal prerequisites are listed, short of a working knowledge of automata theory. However, the material does require a certain level of mathematical maturity. The presentation is in two parts, splitting the book roughly in half.

Part I ("Communicating systems") covers the basic material on communicating system, with a focus on the algebraic treatment of concurrency and communication. There is no mobility in this first part. It summarizes the work of Milner on CCS, explained more thoroughly in say [3]. The presentation is very much geared towards the introduction of mobility in the second part of the monograph. After a quick introductory chapter, Chapter 2 looks at the interaction of automata, and why classical language equivalence for automata is not appropriate when we consider interacting automata. Chapter 3 introduces sequential processes, as a way of describing labelled transition systems, and defines the all-important concept of bisimulation. Chapter 4 extends the notion of processes to concurrent processes. Chapter 5 generalizes the labelled transitions systems and bisimulation results of chapter 3 to the concurrent setting, and introduces reaction rules for concurrent processes (which are reduction rules taking into account communication). Chapters 6 and 7 focus on notions of equivalence which can be established by looking at how a process interacts externally with others (regardless of which internal actions it performs).

Part II of the monograph ("The π-Calculus") introduces mobility into the framework of part I, that is the idea of sending channel names over communication channels to enable communication. Chapter 8 motivates this idea, by

[*] R. Milner, *Communicating and Mobile Systems: The π-calculus*, Cambridge University Press, 1999, 161pp, ISBN 0521658691.

giving examples where this kind of mobility arises naturally in systems. Chapter 9 introduces the π-calculus itself and its reaction rules. Both the monadic version (channels carry a single value at a time) and the polyadic version (many values per channel) are described. Chapter 10 gives sample applications, including how to model simple systems, how to represent data structures, how to program with lists in a functional style. Chapter 11 introduces a simple type system for the calculus, which classifies the kind of information exchanged during an interaction. The type system is used to help design communication protocols to model object-oriented features in the π-calculus, as well as functional programming features. Chapter 12 turns to theory and defines bisimilarity for π-calculus processes. Chapter 13 studies observational equivalence for π-calculus processes. Chapter 14 concludes with a discussion and an overview of related work.

Positive points. The monograph is short, self-contained, and extremely readable. It covers all the important points in enough detail for the subtleties involved to be understood, while still being general enough that the material is applicable to other concurrent calculi variants (especially the material in part I). There are many examples throughout the book, and they form an intrinsic part of the material, serving both a motivating and an illustrative purpose. The sample applications of the π-calculus help the reader make the transition from seeing the π-calculus as an abstract mathematical toy to seeing it as a formalization of a simple concurrent programming language.

Slightly less positive points. The book is short and very focused. Therefore, there is not as much theory as one expects from a book about a formal model of concurrent computation. Moreover, many very interesting venues of investigation are not described (the current work on bisimulation in more general concurrency models, to pick one example, or the variations obtained by considering different semantics for the primitives). In all fairness, the book does not claim to be a theory book, or to cover the broad spectrum of approaches to concurrent calculi, or even to bring the reader up to date with present-day research. Milner refers both to his earlier work on CCS [3] and the forthcoming book by Sangiorgi and Walker "A Theory of Mobile Processes" for a more in-depth treatment of those aspects.

In summary. This monograph is an excellent introduction to the π-calculus. It will not bring much to the researcher already familiar with the π-calculus, save a convenient self-contained reference booklet, but is by far the best starting point for the beginner.

References

[1] C. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.

[2] R. Milner. *A Calculus of Communicating Systems*. Number 92 in Lecture Notes in Computer Science. Springer-Verlag, 1980.

[3] R. Milner. *Communication and Concurrency*. Prentice-Hall, 1989.

[4] R. Milner. The polyadic π-calculus: a tutorial. In W. Brauer F.L. Bauer and H. Schwichtenberg, editors, *Logic and Algebra of Specification*, pages 203–246. Springer-Verlag, 1993.

[5] R. Milner, J. Parrow, and D. Walker. A calculus of mobile processes, parts I and II. *Information and Computation*, 100(1):1–77, 1992.

[6] D. Sangiorgi. *Expressing mobility in process algebras: first-order and higher-order paradigms*. PhD thesis, Department of Computer Science, University of Edinburgh, 1993. CST-99-93, also published as ECS-LFCS-93-266.

[7] D. Walker. Objects in the π-calculus. *Information and Computation*, 115:253–271, 1995.