

Gilles Dowek  
Jean-Jacques Lévy

UNDERGRADUATE TOPICS  
in COMPUTER SCIENCE

# Introduction to the Theory of Programming Languages

 Springer

  
UTiCS

Gilles Dowek · Jean-Jacques Lévy

---

# Introduction to the Theory of Programming Languages



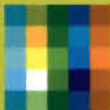
 Springer

# Contents

<b>1</b>	<b>Terms and Relations</b>	1
1.1	Inductive Definitions	1
1.1.1	The Fixed Point Theorem	1
1.1.2	Inductive Definitions	4
1.1.3	Structural Induction	6
1.1.4	The Reflexive-Transitive Closure of a Relation	6
1.2	Languages	7
1.2.1	Languages Without Variables	7
1.2.2	Variables	7
1.2.3	Many-Sorted Languages	9
1.2.4	Free and Bound Variables	10
1.2.5	Substitution	10
1.3	Three Ways to Define the Semantics of a Language	12
1.3.1	Denotational Semantics	12
1.3.2	Big-Step Operational Semantics	12
1.3.3	Small-Step Operational Semantics	12
1.3.4	Non-termination	13
<b>2</b>	<b>The Language PCF</b>	15
2.1	A Functional Language: PCF	15
2.1.1	Programs Are Functions	15
2.1.2	Functions Are First-Class Objects	15
2.1.3	Functions with Several Arguments	16
2.1.4	No Assignments	16
2.1.5	Recursive Definitions	16
2.1.6	Definitions	17
2.1.7	The Language PCF	17
2.2	Small-Step Operational Semantics for PCF	18
2.2.1	Rules	18
2.2.2	Numbers	19
2.2.3	A Congruence	20
2.2.4	An Example	21

2.2.5	Irreducible Closed Terms . . . . .	22
2.2.6	Non-termination . . . . .	23
2.2.7	Confluence . . . . .	24
2.3	Reduction Strategies . . . . .	24
2.3.1	The Notion of a Strategy . . . . .	24
2.3.2	Weak Reduction . . . . .	26
2.3.3	Call by Name . . . . .	26
2.3.4	Call by Value . . . . .	27
2.3.5	A Bit of Laziness Is Needed . . . . .	27
2.4	Big-Step Operational Semantics for PCF . . . . .	27
2.4.1	Call by Name . . . . .	28
2.4.2	Call by Value . . . . .	29
2.5	Evaluation of PCF Programs . . . . .	31
<b>3</b>	<b>From Evaluation to Interpretation . . . . .</b>	<b>33</b>
3.1	Call by Name . . . . .	33
3.2	Call by Value . . . . .	35
3.3	An Optimisation: de Bruijn Indices . . . . .	36
3.4	Construction of Functions via Fixed Points . . . . .	38
3.4.1	First Variation: Recursive Closures . . . . .	38
3.4.2	Second Variation: Rational Values . . . . .	40
<b>4</b>	<b>Compilation . . . . .</b>	<b>43</b>
4.1	An Interpreter Written in a Language Without Functions . . . . .	44
4.2	From Interpretation to Compilation . . . . .	44
4.3	An Abstract Machine for PCF . . . . .	45
4.3.1	The Environment . . . . .	45
4.3.2	Closures . . . . .	46
4.3.3	PCF Constructs . . . . .	46
4.3.4	Using de Bruijn Indices . . . . .	47
4.3.5	Small-Step Operational Semantics . . . . .	48
4.4	Compilation of PCF . . . . .	48
<b>5</b>	<b>PCF with Types . . . . .</b>	<b>51</b>
5.1	Types . . . . .	51
5.1.1	PCF with Types . . . . .	52
5.1.2	The Typing Relation . . . . .	53
5.2	No Errors at Run Time . . . . .	54
5.2.1	Using Small-Step Operational Semantics . . . . .	55
5.2.2	Using Big-Step Operational Semantics . . . . .	55
5.3	Denotational Semantics for Typed PCF . . . . .	56
5.3.1	A Trivial Semantics . . . . .	56
5.3.2	Termination . . . . .	57
5.3.3	Scott's Ordering Relation . . . . .	58
5.3.4	Semantics of Fixed Points . . . . .	59

- 6 Type Inference** . . . . . 63
  - 6.1 Inferring Monomorphic Types . . . . . 63
    - 6.1.1 Assigning Types to Untyped Terms . . . . . 63
    - 6.1.2 Hindley’s Algorithm . . . . . 64
    - 6.1.3 Hindley’s Algorithm with Immediate Resolution . . . . . 66
  - 6.2 Polymorphism . . . . . 68
    - 6.2.1 PCF with Polymorphic Types . . . . . 68
    - 6.2.2 The Algorithm of Damas and Milner . . . . . 70
- 7 References and Assignment** . . . . . 73
  - 7.1 An Extension of PCF . . . . . 74
  - 7.2 Semantics of PCF with References . . . . . 75
- 8 Records and Objects** . . . . . 81
  - 8.1 Records . . . . . 81
    - 8.1.1 Labelled Fields . . . . . 81
    - 8.1.2 An Extension of PCF with Records . . . . . 82
  - 8.2 Objects . . . . . 85
    - 8.2.1 Methods and Functional Fields . . . . . 85
    - 8.2.2 What Is “Self”? . . . . . 86
    - 8.2.3 Objects and References . . . . . 88
- 9 Epilogue** . . . . . 89
- References** . . . . . 93
- Index** . . . . . 95



UNDERGRADUATE TOPICS  
in COMPUTER SCIENCE

**UTiCS** Undergraduate Topics in Computer Science (UTiCS) delivers high-quality instructional content for undergraduates studying in all areas of computing and information science. From core foundational and theoretical material to final-year topics and applications, UTiCS books take a fresh, concise, and modern approach and are ideal for self-study or for a one- or two-semester course. The texts are all authored by established experts in their fields, reviewed by an international advisory board, and contain numerous examples and problems. Many include fully worked solutions.

Gilles Dowek • Jean-Jacques Lévy

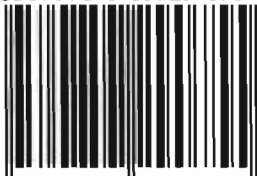
## Introduction to the Theory of Programming Languages

The design and implementation of programming languages, from Fortran and Cobol to Caml and Java, has been one of the key developments in the management of ever more complex computerized systems. *Introduction to the Theory of Programming Languages* gives the reader the means to discover the tools to think, design, and implement these languages.

It proposes a unified vision of the different formalisms that permit definition of a programming language: small steps operational semantics, big steps operational semantics, and denotational semantics, emphasising that all seek to define a relation between three objects: a program, an input value, and an output value. These formalisms are illustrated by presenting the semantics of some typical features of programming languages: functions, recursivity, assignments, records, objects, ... showing that the study of programming languages does not consist of studying languages one after another, but is organized around the features that are present in these various languages. The study of these features leads to the development of evaluators, interpreters and compilers, and also type inference algorithms, for small languages.

COMPUTER SCIENCE

ISBN 978-0-85729-075-5



9 780857 290755

 [springer.com](http://springer.com)

