

Contents

<i>Preface</i>	<i>page ix</i>
Part I: Introduction and Overview	1
1 Introduction	2
1.1 Central Questions	2
1.2 Structure of this Chapter	2
1.3 Basic Concepts of Concurrency	3
1.4 Why Concurrent Programs Should be Proved Correct	8
1.5 The Approach of this Book	33
1.6 Compositionality	46
1.7 From Noncomp. to Comp. Proof Methods – a historical perspective	62
Part II: The Inductive Assertion Method	71
2 Floyd’s Inductive Assertion Method for Transition Diagrams	72
2.1 Objectives of Part II	72
2.2 Structure of this Chapter	75
2.3 Sequential Transition Diagrams and Systems	76
2.4 Specification and Correctness Statements	82
2.5 A Proof Method for Partial Correctness	88
2.6 Soundness	92
2.7 Semantic Completeness of the Inductive Assertion Method	93
2.8 Proving Convergence	98
2.9 Proving Absence of Runtime Errors	104
2.10 Historical Notes	111
3 The Inductive Assertion Method for Shared-Variable Concurrency	119
3.1 Objective and Structure of this Chapter	119
3.2 A Characterisation of Concurrent Execution	121

Cambridge University Press
 978-0-521-80608-4 - Concurrency Verification: Introduction to Compositional and
 Noncompositional Methods
 Willem-Paul de Roever, Frank de Boer, Ulrich Hannemann, Jozef Hooman, Yassine Lakhnech,
 Mannes Poel and Job Zwiers
 Table of Contents
[More information](#)

vi		<i>Contents</i>
3.3	Is this Characterisation of Concurrent Execution Justified?	132
3.4	The Generalisation of Floyd's Approach to Nondeterministic Interleavings	134
3.5	Concurrent Transition Systems with Shared Variables	137
3.6	Proving Convergence for Shared-Variable Concurrency	188
3.7	Proving Deadlock Freedom	203
3.8	Proving Absence of Runtime Errors	206
3.9	Historical Notes	208
4	The Inductive Assertion Method for Synchronous Message Passing	221
4.1	Objective and Introduction	221
4.2	Structure of this Chapter	223
4.3	Syntax and Semantics of Synchronous Transition Diagrams	223
4.4	Proof Methods for Partial Correctness	227
4.5	Semantic Completeness	249
4.6	Technical Note: Modifications Towards Compositionality	264
4.7	A Modular Method for Proving Convergence	269
4.8	Verifying Deadlock Freedom	277
4.9	Proving Absence of Runtime Errors	279
4.10	Historical Notes	282
5	Expressibility and Relative Completeness	291
5.1	Objective	291
5.2	Structure of this Chapter	292
5.3	Syntactic Notions	292
5.4	Partial Correctness of Syntactic Transition Diagrams	298
5.5	Relative Completeness of Floyd's Inductive Assertion Method	300
5.6	Relative Completeness of the Method of Owicki & Gries	309
5.7	Relative Completeness of the Method of Apt, Francez & de Roever	312
5.8	Historical Notes	316
	<i>Picture Gallery</i>	319
	Part III: Compositional Methods based on Assertion Networks	353
6	Introduction to Compositional Reasoning	354
6.1	Motivation	354
6.2	Introduction to Part III and to this Chapter	356
6.3	Assume-Guarantee-based Reasoning	359
6.4	Assumption-Commitment-based Reasoning	361
6.5	Rely-Guarantee-based Reasoning	363

Cambridge University Press
 978-0-521-80608-4 - Concurrency Verification: Introduction to Compositional and
 Noncompositional Methods
 Willem-Paul de Roever, Frank de Boer, Ulrich Hannemann, Jozef Hooman, Yassine Lakhnech,
 Mannes Poel and Job Zwiers
 Table of Contents
[More information](#)

<i>Contents</i>	vii
7 Compositional Proof Methods: Synchronous Message Passing	367
7.1 Objective and Introduction	367
7.2 Structure of the Chapter	368
7.3 Top-level Synchronous Message Passing	369
7.4 A Compositional Proof Method for Nested Parallelism	379
7.5 Assumption-Commitment-based Reasoning	397
7.6 Historical Notes	429
8 Compositional Proof Methods: Shared-Variable Concurrency	438
8.1 Introduction and Overview	438
8.2 Concurrent Transition Diagrams	439
8.3 Top-Level Shared-Variable Concurrency	440
8.4 The Rely-Guarantee Method	447
8.5 Historical Notes	479
Part IV: Hoare Logic	487
9 A Proof System for Sequential Programs Using Hoare Triples	488
9.1 Introduction and Overview of Hoare Logics	488
9.2 Structure of this Chapter	497
9.3 Syntax and Informal Meaning of GCL ⁺ Programs	498
9.4 Semantics of GCL ⁺	501
9.5 A Proof System for GCL ⁺ Programs	506
9.6 Soundness and Relative Completeness	511
9.7 Proof Outlines	517
9.8 Alternative Definitions of Proof Outlines	521
9.9 Examples of Verification during Program Development	522
9.10 Historical Notes	526
10 A Hoare Logic for Shared-Variable Concurrency	531
10.1 Introduction and Overview	531
10.2 Syntax and Informal Meaning of SVL Programs	532
10.3 Semantics of SVL ⁺	537
10.4 A Proof System for SVL Programs	540
10.5 An Extended Example: Concurrent Garbage Collection	563
10.6 Completeness of the Owicki & Gries Method	584
11 A Hoare Logic for Synchronous Message Passing	600
11.1 Structure of this Chapter	600
11.2 Syntax and Informal Meaning of DML Programs	601
11.3 Semantics of DML	606
11.4 A Hoare Logic for Synchronous Message Passing	608
11.5 Soundness and Relative Completeness of this Hoare Logic	630

Cambridge University Press
978-0-521-80608-4 - Concurrency Verification: Introduction to Compositional and
Noncompositional Methods
Willem-Paul de Roever, Frank de Boer, Ulrich Hannemann, Jozef Hooman, Yassine Lakhnech,
Mannes Poel and Job Zwiers
Table of Contents
[More information](#)

viii		<i>Contents</i>
11.6	Technical Note: Modifications Towards Compositionality	638
	Part V: Layered Design	653
12	Transformational Design and Hoare Logic	654
12.1	Introduction and Overview	654
12.2	Structure of this Chapter	660
12.3	Syntax and Informal Meaning of SVL ⁺⁺ Programs	660
12.4	The Semantics of SVL ⁺⁺ Programs	662
12.5	Partial Orders and Temporal Logic	663
12.6	The Communication-Closed-Layers Laws	676
12.7	The Two-Phase Commit Protocol	688
12.8	Assertion-Based Program Transformations	694
12.9	Loop Distribution	696
12.10	Set-partitioning Revisited	700
12.11	Historical Notes	704
	<i>Bibliography</i>	710
	<i>Glossary of Symbols</i>	747
	<i>Index</i>	761