

Krzysztof R. Apt
Frank S. de Boer
Ernst-Rüdiger Olderog

Verification of
Sequential and Concurrent
Programs

Third, Extended Edition

Springer

Contents

<i>Preface</i>	v
Outlines of One-semester Courses	x
Part I In the Beginning	
1 <i>Introduction</i>	3
1.1 An Example of a Concurrent Program	4
Solution 1	4
Solution 2	5
Solution 3	6
Solution 4	8
Solution 5	9
Solution 6	10
1.2 Program Correctness	11
1.3 Structure of this Book	13
1.4 Automating Program Verification	16
1.5 Assertional Methods in Practice	17
2 <i>Preliminaries</i>	19
2.1 Mathematical Notation	21
Sets	21
Tuples	22
Relations	23
Functions	23
Sequences	24
Strings	25
Proofs	26
Induction	27
Grammars	29
2.2 Typed Expressions	29
Types	29

	Variables	30
	Constants	30
	Expressions	31
	Subscripted Variables	32
2.3	Semantics of Expressions	32
	Fixed Structure	33
	States	34
	Definition of the Semantics	35
	Updates of States	36
2.4	Formal Proof Systems	38
2.5	Assertions	39
2.6	Semantics of Assertions	41
2.7	Substitution	42
2.8	Substitution Lemma	47
2.9	Exercises	50
2.10	Bibliographic Remarks	51

Part II Deterministic Programs

3	while Programs	55
3.1	Syntax	57
3.2	Semantics	58
	Properties of Semantics	62
3.3	Verification	63
	Partial Correctness	65
	Total Correctness	70
	Decomposition	73
	Soundness	73
3.4	Proof Outlines	79
	Partial Correctness	79
	Total Correctness	83
3.5	Completeness	85
3.6	Parallel Assignment	91
3.7	Failure Statement	94
3.8	Auxiliary Axioms and Rules	97
3.9	Case Study: Partitioning an Array	99
3.10	Systematic Development of Correct Programs	113
	Summation Problem	115
3.11	Case Study: Minimum-Sum Section Problem	116
3.12	Exercises	121
3.13	Bibliographic Remarks	124

4	<i>Recursive Programs</i>	127
4.1	Syntax	129
4.2	Semantics	129
	Properties of the Semantics	131
4.3	Verification	132
	Partial Correctness	132
	Total Correctness.....	134
	Decomposition	137
	Discussion	138
	Soundness	139
4.4	Case Study: Binary Search	144
	Partial Correctness	145
	Total Correctness.....	147
4.5	Exercises	149
4.6	Bibliographic Remarks	150
5	<i>Recursive Programs with Parameters</i>	151
5.1	Syntax	152
5.2	Semantics	154
5.3	Verification	157
	Partial Correctness: Non-recursive Procedures.....	158
	Partial Correctness: Recursive Procedures	162
	Modularity	165
	Total Correctness.....	165
	Soundness	167
5.4	Case Study: <i>Quicksort</i>	172
	Formal Problem Specification	173
	Properties of <i>Partition</i>	173
	Auxiliary Proof: Permutation Property	174
	Auxiliary Proof: Sorting Property	175
	Total Correctness.....	180
5.5	Exercises	182
5.6	Bibliographic Remarks	182
6	<i>Object-Oriented Programs</i>	185
6.1	Syntax	187
	Local Expressions	187
	Statements and Programs	188
6.2	Semantics	192
	Semantics of Local Expressions.....	192
	Updates of States	194
	Semantics of Statements and Programs.....	195
6.3	Assertions	197
	Substitution	199
6.4	Verification	200

	Partial Correctness	201
	Total Correctness	204
6.5	Adding Parameters	206
	Semantics	207
	Partial Correctness	208
	Total Correctness	210
6.6	Transformation of Object-Oriented Programs	211
	Soundness	214
6.7	Object Creation	217
	Semantics	218
	Assertions	219
	Verification	223
	Soundness	225
6.8	Case Study: Zero Search in Linked List	226
	Partial Correctness	226
	Total Correctness	229
6.9	Case Study: Insertion into a Linked List	232
6.10	Exercises	238
6.11	Bibliographic Remarks	240

Part III Parallel Programs

7	<i>Disjoint Parallel Programs</i>	245
7.1	Syntax	247
7.2	Semantics	248
	Determinism	249
	Sequentialization	252
7.3	Verification	253
	Parallel Composition	254
	Auxiliary Variables	256
	Soundness	259
7.4	Case Study: Find Positive Element	261
7.5	Exercises	264
7.6	Bibliographic Remarks	266
8	<i>Parallel Programs with Shared Variables</i>	267
8.1	Access to Shared Variables	269
8.2	Syntax	270
8.3	Semantics	271
	Atomicity	272
8.4	Verification: Partial Correctness	274
	Component Programs	274
	No Compositionality of Input/Output Behavior	275
	Parallel Composition: Interference Freedom	276
	Auxiliary Variables Needed	279

	Soundness	282
8.5	Verification: Total Correctness	284
	Component Programs	284
	Parallel Composition: Interference Freedom	286
	Soundness	288
	Discussion	289
8.6	Case Study: Find Positive Element More Quickly	291
8.7	Allowing More Points of Interference	294
8.8	Case Study: Parallel Zero Search	299
	Step 1. Simplifying the program	299
	Step 2. Proving partial correctness	300
8.9	Exercises	303
8.10	Bibliographic Remarks	305
9	<i>Parallel Programs with Synchronization</i>	307
9.1	Syntax	309
9.2	Semantics	310
9.3	Verification	311
	Partial Correctness	311
	Weak Total Correctness	313
	Total Correctness	314
	Soundness	316
9.4	Case Study: Producer/Consumer Problem	319
9.5	Case Study: The Mutual Exclusion Problem	324
	Problem Formulation	324
	Verification	326
	A Busy Wait Solution	327
	A Solution Using Semaphores	331
9.6	Allowing More Points of Interference	334
9.7	Case Study: Synchronized Zero Search	335
	Step 1. Simplifying the Program	336
	Step 2. Decomposing Total Correctness	337
	Step 3. Proving Termination	337
	Step 4. Proving Partial Correctness	342
9.8	Exercises	344
9.9	Bibliographic Remarks	345
 Part IV Nondeterministic and Distributed Programs		
10	<i>Nondeterministic Programs</i>	349
10.1	Syntax	351
10.2	Semantics	352
	Properties of Semantics	353
10.3	Why Are Nondeterministic Programs Useful?	354
	Symmetry	355

	Nondeterminism	355
	Failures	356
	Modeling Concurrency	356
10.4	Verification	357
	Partial Correctness	357
	Total Correctness	357
	Soundness	359
10.5	Case Study: The Welfare Crook Problem	360
10.6	Transformation of Parallel Programs	363
10.7	Exercises	368
10.8	Bibliographic Remarks	370
11	<i>Distributed Programs</i>	373
11.1	Syntax	375
	Sequential Processes	375
	Distributed Programs	376
11.2	Semantics	380
11.3	Transformation into Nondeterministic Programs	382
	Semantic Relationship Between S and $T(S)$	382
	Proof of the Sequentialization Theorem	385
11.4	Verification	390
	Partial Correctness	390
	Weak Total Correctness	391
	Total Correctness	391
	Proof Systems	392
	Soundness	393
11.5	Case Study: A Transmission Problem	396
	Step 1. Decomposing Total Correctness	397
	Step 2. Proving Partial Correctness	397
	Step 3. Proving Absence of Failures and of Divergence	399
	Step 4. Proving Deadlock Freedom	400
11.6	Exercises	402
11.7	Bibliographic Remarks	405
12	<i>Fairness</i>	407
12.1	The Concept of Fairness	409
	Selections and Runs	410
	Fair Nondeterminism Semantics	412
12.2	Transformational Semantics	413
12.3	Well-Founded Structures	413
12.4	Random Assignment	414
	Semantics	415
	Verification	415
12.5	Schedulers	419
	The Scheduler FAIR	421

	The Scheduler RORO	424
	The Scheduler QUEUE	426
12.6	Transformation	427
12.7	Verification	430
	Total Correctness.....	430
	Soundness	438
12.8	Case Study: Zero Search	442
12.9	Case Study: Asynchronous Fixed Point Computation	446
12.10	Exercises	452
12.11	Bibliographic Remarks	455
A	<i>Semantics</i>	457
B	<i>Axioms and Proof Rules</i>	459
C	<i>Proof Systems</i>	471
D	<i>Proof Outlines</i>	475
	<i>References</i>	477
	<i>Index</i>	491
	<i>Author Index</i>	496
	<i>Symbol Index</i>	501