## Contents

1. **Preface**
   - xi

2. **Program Analysis**
   - 1
     - 1.1 Understanding Software Behavior
     - 1.2 Program Analysis Applications and Challenges
     - 1.3 Concepts in Program Analysis
       - 1.3.1 What to Analyze
       - 1.3.2 Static versus Dynamic
       - 1.3.3 A Hard Limit: Uncomputability
       - 1.3.4 Automation and Scalability
       - 1.3.5 Approximation: Soundness and Completeness
     - 1.4 Families of Program Analysis Techniques
       - 1.4.1 Testing: Checking a Set of Finite Executions
       - 1.4.2 Assisted Proof: Relying on User-Supplied Invariants
       - 1.4.3 Model Checking: Exhaustive Exploration of Finite Systems
       - 1.4.4 Conservative Static Analysis: Automatic, Sound, and Incomplete Approach
       - 1.4.5 Bug Finding: Error Search, Automatic, Unsound, Incomplete, Based on Heuristics
     - 1.5 Roadmap
     - 17

3. **A Gentle Introduction to Static Analysis**
   - 19
     - 2.1 Semantics and Analysis Goal: A Reachability Problem
     - 2.2 Abstraction
     - 2.3 A Computable Abstract Semantics: Compositional Style
       - 2.3.1 Abstraction of Initialization
       - 2.3.2 Abstraction of Post-Conditions
       - 2.3.3 Abstraction of Non-Deterministic Choice
       - 2.3.4 Abstraction of Non-Deterministic Iteration
       - 2.3.5 Verification of the Property of Interest
     - 2.4 A Computable Abstract Semantics: Transitional Style
     - 44
## Contents

2.4.1 Semantics as State Transitions | 45  
2.4.2 Abstraction of States | 48  
2.4.3 Abstraction of State Transitions | 48  
2.4.4 Analysis by Global Iterations | 51  
2.5 Core Principles of a Static Analysis | 56  

3 A General Static Analysis Framework Based on a Compositional Semantics | 59  
3.1 Semantics | 60  
3.1.1 A Simple Programming Language | 60  
3.1.2 Concrete Semantics | 60  
3.2 Abstractions | 65  
3.2.1 The Concept of Abstraction | 65  
3.2.2 Non-Relational Abstraction | 69  
3.2.3 Relational Abstraction | 72  
3.3 Computable Abstract Semantics | 74  
3.3.1 Abstract Interpretation of Assignment | 76  
3.3.2 Abstract Interpretation of Conditional Branching | 79  
3.3.3 Abstract Interpretation of Loops | 82  
3.3.4 Putting Everything Together | 90  
3.4 The Design of an Abstract Interpreter | 92  

4 A General Static Analysis Framework Based on a Transitional Semantics | 95  
4.1 Semantics as State Transitions | 96  
4.1.1 Concrete Semantics | 96  
4.1.2 Recipe for Defining a Concrete Transitional Semantics | 100  
4.2 Abstract Semantics as Abstract State Transitions | 101  
4.2.1 Abstraction of the Semantic Domain | 102  
4.2.2 Abstraction of Semantic Functions | 105  
4.2.3 Recipe for Defining an Abstract Transition Semantics | 107  
4.3 Analysis Algorithms Based on Global Iterations | 109  
4.3.1 Basic Algorithms | 109  
4.3.2 Worklist Algorithm | 110  
4.4 Use Example of the Framework | 112  
4.4.1 Simple Imperative Language | 112  
4.4.2 Concrete State Transition Semantics | 114  
4.4.3 Abstract State | 115  
4.4.4 Abstract State Transition Semantics | 116  

5 Advanced Static Analysis Techniques | 119  
5.1 Construction of Abstract Domains | 120  
5.1.1 Abstraction of Boolean-Numerical Properties | 120  
5.1.2 Describing Conjunctive Properties | 122  
5.1.3 Describing Properties Involving Case Splits | 126  
5.1.4 Construction of an Abstract Domain | 130  
5.2 Advanced Iteration Techniques | 131
## Contents

5.2.1 Loop Unrolling 131  
5.2.2 Fixpoint Approximation with More Precise Widening Iteration 133  
5.2.3 Refinement of an Abstract Approximation of a Least Fixpoint 135  

5.3 Sparse Analysis 137  
5.3.1 Exploiting Spatial Sparsity 139  
5.3.2 Exploiting Temporal Sparsity 140  
5.3.3 Precision-Preserving Def-Use Chain by Pre-Analysis 142  

5.4 Modular Analysis 143  
5.4.1 Parameterization, Summary, and Scalability 144  
5.4.2 Case Study 145  

5.5 Backward Analysis 147  
5.5.1 Forward Semantics and Backward Semantics 147  
5.5.2 Backward Analysis and Applications 148  
5.5.3 Precision Refinement by Combined Forward and Backward Analysis 150  

6 Practical Use of Static Analysis Tools 153  
6.1 Analysis Assumptions and Goals 153  
6.2 Setting Up the Static Analysis of a Program 159  
6.2.1 Definition of the Source Code and Proof Goals 159  
6.2.2 Parameters to Guide the Analysis 162  
6.3 Inspecting Analysis Results 166  
6.4 Deployment of a Static Analysis Tool 170  

7 Static Analysis Tool Implementation 173  
7.1 Concrete Semantics and Concrete Interpreter 174  
7.2 Abstract Domain Implementation 180  
7.3 Static Analysis of Expressions and Conditions 183  
7.4 Static Analysis Based on a Compositional Semantics 186  
7.5 Static Analysis Based on a Transitional Semantics 190  

8 Static Analysis for Advanced Programming Features 195  
8.1 For a Language with Pointers and Dynamic Memory Allocations 196  
8.1.1 Language and Concrete Semantics 196  
8.1.2 An Abstract Semantics 199  
8.2 For a Language with Functions and Recursive Calls 203  
8.2.1 Language and Concrete Semantics 203  
8.2.2 An Abstract Semantics 206  
8.3 Abstractions for Data Structures 212  
8.3.1 Arrays 212  
8.3.2 Buffers and Strings 217  
8.3.3 Pointers 219  
8.3.4 Dynamic Heap Allocation 223  
8.4 Abstraction for Control Flow Structures 229  
8.4.1 Functions and Procedures 230  
8.4.2 Parallelism 237
9 Classes of Semantic Properties and Verification by Static Analysis

9.1 Trace Properties
  9.1.1 Safety
  9.1.2 Liveness
  9.1.3 General Trace Properties

9.2 Beyond Trace Properties: Information Flows and Other Properties

10 Specialized Static Analysis Frameworks

10.1 Static Analysis by Equations
  10.1.1 Data-Flow Analysis

10.2 Static Analysis by Monotonic Closure
  10.2.1 Pointer Analysis
  10.2.2 Higher-Order Control-Flow Analysis

10.3 Static Analysis by Proof Construction
  10.3.1 Type Inference

11 Summary and Perspectives

A Reference for Mathematical Notions and Notations

A.1 Sets
A.2 Logical Connectives
A.3 Definitions and Proofs by Induction
A.4 Functions
A.5 Order Relations and Ordered Sets
A.6 Operators over Ordered Structures and Fixpoints

B Proofs of Soundness

B.1 Properties of Galois Connections
B.2 Proofs of Soundness for Chapter 3
  B.2.1 Soundness of the Abstract Interpretation of Expressions
  B.2.2 Soundness of the Abstract Interpretation of Conditions
  B.2.3 Soundness of the Abstract Join Operator
  B.2.4 Abstract Iterates with Widening
  B.2.5 Soundness of the Abstract Interpretation of Commands
B.3 Proofs of Soundness for Chapter 4
  B.3.1 Transitional-Style Static Analysis over Finite-Height Domains
  B.3.2 Transitional-Style Static Analysis with Widening
  B.3.3 Use Example of the Transitional-Style Static Analysis

Bibliography

Index