Principles and Specifications of Concurrent Systems
Leslie Lamport
Version of 20 August 2015

The Principles and Specification Tracks

1 Introduction
   1.1 Concurrent Computation
   1.2 Modeling Computation
   1.3 Specification
   1.4 Systems and Languages

2 The One-Bit Clock
   2.1 The Clock’s Behaviors
   2.2 Describing the Behaviors
   2.3 Writing the Specification
   2.4 The Pretty-Printed Version of Your Spec
   2.5 Checking the Specification
   2.6 Computing the Behaviors from the Specification
   2.7 Other Ways of Writing the Behavior Specification
   2.8 Specifying the Clock in PlusCal

3 The Die Hard Problem
   3.1 Representing the Problem in TLA+
   3.2 Applying TLC
   3.3 Expressing the Problem in PlusCal

4 Euclid’s Algorithm
   4.1 The Greatest Common Divisor
      4.1.1 Divisors
      4.1.2 choose and the Maximum of a Set
      4.1.3 The GCD Operator
   4.2 Comments
   4.3 The Algorithm
   4.4 The TLA+ Translation
   4.5 Checking Safety
   4.6 Checking Liveness
   4.7 The Translation Revisited
   4.8 The Grain of Atomicity
4.9 Why Euclid’s Algorithm Is Correct
   4.9.1 Proving Invariance
   4.9.2 Verifying $GCD_1-GCD_3$
   4.9.3 Proving Termination

4.10 Euclid’s Algorithm for Sets

5 The Generalized Die Hard Problem
   5.1 The PlusCal Representation
   5.2 Checking the Algorithm
   5.3 The TLA+ Translation

6 Alternation
   6.1 The Problem
   6.2 The One-Bit Clock Revisited
   6.3 Specifying Alternation: Safety
   6.4 Specifying Alternation: Liveness
   6.5 The Two-Phase Handshake Protocol
   6.6 Refinement
   6.7 Refinement and Stuttering
      6.7.1 Adding Steps
      6.7.2 Temporal Logic and Stuttering
      6.7.3 A Finer-Grained Algorithm
   6.8 Temporal Logic and Refinement
   6.9 Alternation Revisited
   6.10 Round-Robin Synchronization
      6.10.1 The One-Bit Clock Revisited Again
      6.10.2 An $N$-Valued Clock
      6.10.3 An Implementation of the $N$-Valued Clock
      6.10.4 Round-Robin Synchronization

The Principles Track

7 Mutual Exclusion
   7.1 The Problem
   7.2 The One-Bit Protocol
      7.2.1 The Protocol
      7.2.2 An Assertional Proof
      7.2.3 Using TLC to Check an Inductive Invariant
   7.3 The Two-Process One-Bit Algorithm
      7.3.1 The Two-Process Algorithm
      7.3.2 Busy Waiting Versus Synchronization Primitives
      7.3.3 Requirement (c)
   7.4 Proving Liveness
7.5 An Informal Proof
7.6 A More Formal Proof
7.7 The $N$-Process One-Bit Algorithm
7.8 The Bakery Algorithm
  7.8.1 The Big-Step Algorithm
  7.8.2 Choosing the Grain of Atomicity
  7.8.3 The Atomic Bakery Algorithm
  7.8.4 The Real Bakery Algorithm
7.9 Mutual Exclusion in Modern Programs

8 The Bounded Channel and Bounded Buffer
  8.1 The Bounded Channel
    8.1.1 The Specification
    8.1.2 Safety
    8.1.3 Liveness
    8.1.4 Implementing The Bounded Channel
  8.2 The Bounded Buffer
    8.2.1 Modular Arithmetic
    8.2.2 The Algorithm
  8.3 The Bounded Buffer Implements the Bounded Channel
    8.3.1 The Refinement Mapping
    8.3.2 Showing Implementation
    8.3.3 Liveness
  8.4 A Finer-Grained Bounded Buffer
  8.5 Further Refinement
  8.6 What is a Process?

The Specification Track

9 An Input/Output Specification
  9.1 The Example
  9.2 Sorting
  9.3 Votes
  9.4 The Borda Ranking
  9.5 The Condorcet Ranking
  9.6 Transitive Closure
    9.6.1 A Mathematical Definition
    9.6.2 A Definition TLC Can Execute Faster
    9.6.3 Warshall's Algorithm
  9.7 The Condorcet Ranking Revisited
The TLA+ Proof Track

10 About Proofs and Proving
   10.1 About Proofs
   10.2 About TLAPS

11 Correctness of Euclid’s Algorithm
   11.1 Proving Safety
   11.2 Proving Properties of the GCD

12 The Proof Language
   12.1 What a Theorem Asserts
   12.2 The Hierarchical Structure of a Proof
      12.2.1 Writing Structured Proofs
      12.2.2 Reading Structured Proofs
   12.3 The State of a Proof
      12.3.1 Steps That Can Have a Proof
      12.3.2 Steps That Cannot Have a Proof
   12.4 Proof Obligations
   12.5 Further Details
      12.5.1 Additional Language Features
      12.5.2 Importing
      12.5.3 Recursively Defined Functions and Operators
      12.5.4 The Fine Print

13 The Bounded Buffer Proof

Math

13 Arithmetic and Logic
   13.1 Arithmetic
   13.2 Mathematical Logic
   13.3 Propositional Logic
      13.3.1 ∧ and ∨
      13.3.2 Other Propositional Operators
   13.4 Predicate Logic
   13.5 The CHOOSE Operator

14 Sets
   14.1 An Introduction to Sets
   14.2 Simple Set Operators
   14.3 Set Constructors
   14.4 SUBSET and UNION
   14.5 Collections too Big to Be Sets
14.6 Bags

15 Functions
15.1 Functions and Their Domains
15.2 Writing Functions
15.3 Sets of Functions
15.4 The EXCEPT Construct
15.5 Tuples
15.6 Records
15.7 Strings

16 Miscellaneous Constructs
16.1 Conditional Constructs
   16.1.1 IF / THEN / ELSE
   16.1.2 CASE
16.2 Definitions
   16.2.1 Simple Operator Definitions
   16.2.2 Function Definitions
   16.2.3 Recursive Operator Definitions
   16.2.4 Recursive Or Inductive?
16.3 The LET / IN Construct
16.4 The LAMBDA Construct

17 Temporal Logic
17.1 Understanding Temporal Formulas
17.2 Proof Rules and Proofs
17.3 Rules for Proving Safety
17.4 Leads To
   17.4.1 The Leads-To Induction Rule
   17.4.2 The □ ~ Rule
   17.4.3 Proving ~ Formulas by Contradiction
17.5 Fairness
   17.5.1 The ENABLED Operator
   17.5.2 Weak Fairness
   17.5.3 Strong Fairness
   17.5.4 Proving ~ Properties with Fairness
   17.5.5 Proving Fairness

Topics

18 Variable Hiding and Auxiliary Variables

19 Reduction
20 Debugging With TLC

20.1 Print Statements
20.2 Having TLC Set and Read Values
20.3 Using LET
20.4 The Perils of Lazy Evaluation