
Appendix A

Mathematical Preliminaries

This appendix gives some necessary background to the mathematical concepts used in this book, many of which are only needed in Chapter 7.

Sets. A *set* is a finite or infinite collection of elements. Examples of sets are the natural numbers $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ and the set $\{a, b, c\}$. We write $a \in A$ if a is an element in the set A . If A_1, \dots, A_n are sets, then $A_1 \times \dots \times A_n$ is another set, the *product* of those sets, whose elements are n -tuples (a_1, \dots, a_n) , where each $a_i \in A_i$. The *powerset* $\mathcal{P}(A)$ of A is the set whose elements are the *subsets* of A .

Functions. Given two sets A and B , a *function* $f : A \rightarrow B$ assigns to *each* element $a \in A$ a *unique* element $f(a) \in B$. It follows from this definition that: if $a_1 = a_2$, then $f(a_1) = f(a_2)$. Any function f is *total*: it can be applied to any value a in A ; and the result $f(a)$ must be defined and must be an element in the set B . A function f that cannot be applied to *all* elements in A is called a *partial* function.

To avoid having to find a name for a function, a function can also be defined using “lambda notation.” For example, $\lambda m, n . m + 2n$ denotes some function $h : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ defined by $h(m, n) = m + 2n$. A function can also be defined by showing the mapping of elements explicitly; for example, $\{a \mapsto 1, b \mapsto 3, c \mapsto 2\}$ defines some function $f_1 : \{a, b, c\} \rightarrow \{1, 2, 3\}$.

The *composition* $g \circ f$ of two functions $f : A \rightarrow B$ and $g : B \rightarrow C$ is a function $g \circ f : A \rightarrow C$ defined by $(g \circ f)(a) = g(f(a))$ for each $a \in A$.

A function $f : A \rightarrow B$ is *surjective* if for *each* element $b \in B$ there is *some* $a \in A$ such that $f(a) = b$. That is, the function f can “reach” all elements in the set B . For example, the above function h is surjective. The function $k : \mathbb{N} \rightarrow \mathbb{N}$ defined by $k(n) = 2n$ is *not* surjective, since the numbers 1, 3, 5, ... cannot be reached by k .

A function $f : A \rightarrow B$ is *injective* if $a_1 \neq a_2$ implies $f(a_1) \neq f(a_2)$; that is, f does not map different A -values to the same value in B . The above function h is *not* injective, since $(4, 2) \neq (2, 3)$, yet $h(4, 2) = h(2, 3) = 8$, whereas k is injective, since $m \neq n$ implies $k(m) = 2m \neq 2n = k(n)$.

A function that is both surjective and injective is *bijective*. A bijective function $f : A \rightarrow B$ has an *inverse* function $f^{-1} : B \rightarrow A$ such that $f^{-1} \circ f$ is the identity function id_A on A , and $f \circ f^{-1}$ is the identity function id_B on B , where, for any set S , the identity function $id_S : S \rightarrow S$ is defined by $id_S(s) = s$ for each $s \in S$. For example, the above function f_1 is bijective, and its inverse is the function $f_1^{-1} : \{1, 2, 3\} \rightarrow \{a, b, c\}$ with $f_1^{-1}(1) = a$, $f_1^{-1}(2) = c$, and $f_1^{-1}(3) = b$.

Relations. A *relation* R over a set S is a subset $R \subseteq S$. We usually write $R(s)$ for $s \in R$. If S is the product $A \times A$, then R is called a *binary relation* over A , and we often write $a_1 R a_2$ for $(a_1, a_2) \in R$.

Partial Orders. A binary relation R over A is a *partial order* (over A) if and only if it satisfies the following properties for all a, b, c in A :

- *Reflexivity:* aRa holds for all $a \in A$.
- *Antisymmetry:* If aRb and bRa both hold, then $a = b$.
- *Transitivity:* If aRb and bRc both hold, then aRc also holds.

Examples of partial orders include the relations \leq and \geq on numbers, and the (reflexive) subset relation \subseteq on sets. The relations $<$ and $>$ on numbers (do not satisfy reflexivity), and “brother of” (does not satisfy antisymmetry) are *not* partial orders.

A binary relation R over A is *strict partial order* if and only if it satisfies:

- *Irreflexivity:* There is no $a \in A$ such that aRa .
- *Transitivity:* Defined as above.

Examples of strict partial orders are the relations $<$ and $>$ on numbers, the “forefather of” relation, and the proper subset relation \subset on sets. Antisymmetry is not mentioned, since aRb and bRa cannot both hold in a strict partial order (why not?).

Equivalence and Congruence Relations. An *equivalence relation* \approx over a set A is a binary relation over A that satisfies the following properties for all $a, b, c \in A$:

- *Reflexivity:* $a \approx a$ for all $a \in A$;
- *Symmetry:* If $a \approx b$ holds, then $b \approx a$ also holds.
- *Transitivity:* If $a \approx b$ and $b \approx c$ both hold, then $a \approx c$ also holds.

Examples of equivalence relations include:

1. Standard equality $=$ over the natural numbers (or any other set for that matter).
2. The relation \equiv_k over \mathbb{N} , for any $k > 1$, defined by $m \equiv_k n$ if and only if $m \bmod k = n \bmod k$ (i.e., m and n have the same remainder when divided by k).
3. The relation \equiv_{card} defined on sets of natural numbers by $s_1 \equiv_{card} s_2$ if and only if s_1 and s_2 have the same cardinality (the same number of distinct elements).
4. The relation *sameFather* on persons, where *sameFather*(p_1, p_2) holds if and only if p_1 and p_2 have the same father.

The relation \leq on natural numbers is not an equivalence relation, since symmetry does not hold: $5 \leq 8$ holds, but not the symmetric $8 \leq 5$.

The *equivalence class* $[a]_{\approx}$ of a w.r.t. the equivalence relation \approx is the set of elements that are \approx -equivalent to a . Formally, $[a]_{\approx} = \{x \in A \mid x \approx a\}$. For example, the equivalence classes over the relation \equiv_3 are

$$\begin{aligned}
 [0]_{\equiv_3} &= \{0, 3, 6, 9, \dots\} \\
 [1]_{\equiv_3} &= \{1, 4, 7, 10, \dots\} \\
 [2]_{\equiv_3} &= \{2, 5, 8, 11, \dots\},
 \end{aligned}$$

and some equivalence classes over the relation \equiv_{card} are:

$$\begin{aligned}
 [\emptyset]_{\equiv_{card}} &= \{\emptyset\} \\
 [\{8\}]_{\equiv_{card}} &= \{\{0\}, \{1\}, \{2\}, \{3\}, \dots\} \\
 [\{4, 5\}]_{\equiv_{card}} &= \{\{0, 1\}, \{0, 2\}, \{0, 3\}, \{1, 2\}, \{2, 3\}, \{1, 4\}, \dots\} \\
 \vdots & \qquad \qquad \qquad \vdots
 \end{aligned}$$

The equivalence classes of an equivalence relation \approx on A partition the set A .

An equivalence relation \approx is a *congruence* on A w.r.t. to a set of functions F (on A) if and only if for each function $f \in F$: if $a_1 \approx a'_1, \dots, a_n \approx a'_n$ all hold, then $f(a_1, \dots, a_n) \approx f(a'_1, \dots, a'_n)$ also holds. For example:

1. Standard equality $=$ is a congruence for any function: If a_1 and a'_1 are the same element, then, by the definition of a function, $f(a_1)$ and $f(a'_1)$ must also be the same element.
2. The relation \equiv_3 is congruence w.r.t. the functions $+$, $-$, and $*$ (prove it!).
3. The relation \equiv_{card} is not a congruence w.r.t. standard set operators such as union and intersection, since $\{1, 2\} \equiv_{card} \{3, 4\}$ and $\{1, 2, 3\} \equiv_{card} \{7, 8, 9\}$, whereas $\{1, 2\} \cup \{1, 2, 3\} \not\equiv_{card} \{3, 4\} \cup \{7, 8, 9\}$.
4. The relation *sameFather* is not congruent w.r.t. the function *mother* : $Person \rightarrow Person$, since even though Aphrodite and Apollo have the same father, Zeus, their respective mothers Dione and Leto do not have the same fathers.

Mathematical Induction. Let $P(n)$ be a property about a natural number n . If you can prove the following:

- Basis: $P(0)$ holds, and
- Induction step: $P(k + 1)$ holds, for *any* natural number k , assuming that $P(k)$ holds (the assumption that $P(k)$ holds is called the *induction hypothesis*).

Then you have proved that $P(n)$ holds for all natural numbers $n \in \{0, 1, 2, \dots\}$.

To prove that $P(n)$ holds for all $n \geq m$, the base case amounts to proving $P(m)$.

Another (equivalent) version of mathematical induction is: If for *any* natural number k , the property $P(k)$ holds when you can assume (as induction hypothesis) $P(k')$ for all $k' < k$, then $P(n)$ holds for all natural numbers n .

Exercise 249 Give an example of a function for which \equiv_3 is not a congruence.

Exercise 250 Prove that $0 + 1 + 2 + \dots + n = \frac{n \cdot (n+1)}{2}$ for all $n \in \mathbb{N}$.

Exercise 251 Prove that $n! \geq 2^n$ for all natural numbers $n \geq 4$.

Exercise 252 Show that the two versions of the induction principle for the natural numbers are equivalent.

References

1. G. Agha, J. Meseguer, and K. Sen. PMAude: Rewrite-based specification language for probabilistic object systems. *Electronic Notes in Theoretical Computer Science*, 153(2):213–239, 2006.
2. M. AlTurki and J. Meseguer. PVESTA: A parallel statistical model checking and quantitative analysis tool. In *Proc. Algebra and Coalgebra in Computer Science (CALCO 2011)*, volume 6859 of *Lecture Notes in Computer Science*. Springer, 2011.
3. M. AlTurki, J. Meseguer, and C. A. Gunter. Probabilistic modeling and analysis of DoS protection for the ASV protocol. *Electronic Notes in Theoretical Computer Science*, 234:3–18, 2009.
4. R. Alur and T. A. Henzinger. Logics and models of real time: A survey. In *Real-Time: Theory in Practice*, volume 600 of *Lecture Notes in Computer Science*. Springer, 1992.
5. A. Armando et al. The AVISPA tool for the automated validation of internet security protocols and applications. In *Proc. Computer Aided Verification (CAV 2005)*, volume 3576 of *Lecture Notes in Computer Science*. Springer, 2005.
6. F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, 1998.
7. K. Bae and J. Meseguer. Model checking linear temporal logic of rewriting formulas under localized fairness. *Science of Computer Programming*, 99:193–234, 2015.
8. C. Baier, J.-P. Katoen, and H. Hermanns. Approximate symbolic model checking of continuous-time Markov chains. In *Proc. Concurrency Theory (CONCUR 1999)*, volume 1664 of *Lecture Notes in Computer Science*. Springer, 1999.
9. J. Baker et al. Megastore: Providing scalable, highly available storage for interactive services. In *Proc. Innovative Data Systems Research (CIDR 2011)*. www.cidrdb.org, 2011.
10. D. Benavay, D. Kapur, and P. Narendran. Complexity of matching problems. In *Proc. Rewriting Techniques and Applications (RTA 1985)*, volume 202 of *Lecture Notes in Computer Science*. Springer, 1985.
11. J. A. Bergstra and J. V. Tucker. A characterization of computable data types by means of a finite, equational specification method. CWI Technical Report IW 124/79, Stichting Mathematisch Centrum, Amsterdam, 1979.
12. J. A. Bergstra and J. V. Tucker. Algebraic specification of computable and semicomputable data types. *Theoretical Computer Science*, 50:137–181, 1987.

13. B. Blanchet. Automatic verification of security protocols in the symbolic model: The verifier ProVerif. In *Foundations of Security Analysis and Design VII (FOSAD 2012/2013)*, volume 8604 of *Lecture Notes in Computer Science*. Springer, 2014.
14. D. Bogdanas and G. Rosu. K-Java: A complete semantics of Java. In *Proc. Principles of Programming Languages (POPL 2015)*. ACM, 2015.
15. E. A. Brewer. Towards robust distributed systems (abstract). In *Proc. Principles of Distributed Computing (PODC 2000)*. ACM, 2000.
16. R. Bruni and J. Meseguer. Semantic foundations for generalized rewrite theories. *Theoretical Computer Science*, 360(1-3):386–414, 2006.
17. M. Burrows, M. Abadi, and R. M. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.
18. E. Chang and R. Roberts. An improved algorithm for decentralized extrema-finding in circular configurations of processes. *Communications of the ACM*, 22:281–283, 1979.
19. S. Chen, J. Meseguer, R. Sasse, H. J. Wang, and Y.-M. Wang. A systematic approach to uncover security flaws in GUI logic. In *Proc. IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2007.
20. M. Clavel, F. Durán, S. Eker, S. Escobar, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. *Maude Manual (Version 2.7.1)*, July 2016. <http://maude.cs.illinois.edu>.
21. M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. Talcott. *All About Maude – A High-Performance Logical Framework*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007.
22. S. A. Cook. The complexity of theorem-proving procedures. In *Proc. ACM Symposium on Theory of Computing (STOC 1971)*. ACM, 1971.
23. G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems: Concepts and Design*. Addison-Wesley, third edition, 2001.
24. C. J. F. Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Proc. Computer Aided Verification (CAV 2008)*, volume 5123 of *Lecture Notes in Computer Science*. Springer, 2008.
25. M. Davis, Y. Matijasevič, and J. Robinson. Hilbert’s tenth problem. Diophantine equations: positive aspects of a negative solution. In *Mathematical Developments Arising from Hilbert Problems, Part 2*, volume 28.2 of *Proceedings of Symposia in Pure Mathematics*. American Mathematical Society, 1976.
26. N. Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17:279–301, 1982.
27. N. Dershowitz. Termination of rewriting. *Journal of Symbolic Computation*, 3:69–116, 1987.
28. W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22:644–654, 1976.
29. E. W. Dijkstra. Two starvation free solutions to a general exclusion problem. EWD 625, Plataanstraat 5, 5671 Al Nuenen, The Netherlands, 1978.
30. D. Dolev and A. Yao. On the security of public-key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.
31. G. Dowek, C. A. Muñoz, and C. Rocha. Rewriting logic semantics of a plan execution language. In *Proc. Structural Operational Semantics (SOS 2009)*, volume 18 of *Electronic Proceedings in Theoretical Computer Science*, 2009.
32. F. Durán, S. Lucas, C. Marché, J. Meseguer, and X. Urbain. Proving operational termination of membership equational programs. *Higher-Order and Symbolic Computation*, 21(1-2):59–88, 2008.
33. F. Durán and J. Meseguer. On the Church-Rosser and coherence properties of conditional order-sorted rewrite theories. *Journal of Logic and Algebraic Programming*, 81(7-8):816–850, 2012.

34. J. Eckhardt, T. Mühlbauer, M. AlTurki, J. Meseguer, and M. Wirsing. Stable availability under denial of service attacks through formal patterns. In *Proc. Fundamental Approaches to Software Engineering (FASE 2012)*, volume 7212 of *Lecture Notes in Computer Science*. Springer, 2012.
35. H. Ehrig and B. Mahr. *Fundamentals of Algebraic Specifications I, Equations and Initial Semantics*, volume 6 of *EATCS Monographs on Theoretical Computer Science*. Springer, 1985.
36. S. Eker. Fast matching in combinations of regular equational theories. *Electronic Notes in Theoretical Computer Science*, 4:90–109, 1996.
37. S. Eker, M. Knapp, K. Laderoute, P. Lincoln, J. Meseguer, and K. Sonmez. Pathway logic: Symbolic analysis of biological signaling. In *Proc. Pacific Symposium on Biocomputing, Hawaii*, Jan 2002.
38. S. Eker, M. Knapp, K. Laderoute, P. Lincoln, and C. Talcott. Pathway logic: Executable models of biological networks. *Electronic Notes in Theoretical Computer Science*, 71:144–161, 2002.
39. C. Ellison and G. Rosu. An executable formal semantics of C with applications. In *Proc. Principles of Programming Languages (POPL 2012)*. ACM, 2012.
40. R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. Addison-Wesley, sixth edition, 2011.
41. E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B. Elsevier, 1990.
42. S. Escobar, C. A. Meadows, and J. Meseguer. Maude-NPA: Cryptographic protocol analysis modulo equational properties. In *Foundations of Security Analysis and Design V, (FOSAD 2007/2008/2009)*, volume 5705 of *Lecture Notes in Computer Science*. Springer, 2009.
43. A. Farzan, F. Chen, J. Meseguer, and G. Rosu. Formal analysis of Java programs in JavaFAN. In *Proc. Computer Aided Verification (CAV 2004)*, volume 3114 of *Lecture Notes in Computer Science*. Springer, 2004.
44. M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32(2):374–382, 1985.
45. N. Francez. *Fairness*. Springer, 1986.
46. H. Garcia-Molina. Elections in distributed computer systems. *IEEE Transactions on Computers*, C-31(1):48–59, 1982.
47. M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman and Company, 1979.
48. GMP home-page. <http://www.swox.com/gmp/>.
49. J. Goguen, J. Thatcher, E. Wagner, and J. Wright. Abstract data types as initial algebras and the correctness of data representations. In *Computer Graphics, Pattern Recognition, and Data Structure*, pages 89–93. IEEE, 1975.
50. J. A. Goguen and J. Meseguer. Order-sorted algebra I: equational deduction for multiple inheritance, overloading, exceptions and partial operations. *Theoretical Computer Science*, 105:217–273, 1992.
51. A. Goodloe, C. A. Gunter, and M.-O. Stehr. Formal prototyping in early stages of protocol design. In *Proc. Issues in the Theory of Security (WITS 2005)*. ACM, 2005.
52. M. T. Goodrich and R. Tamassia. *Data Structures and Algorithms in JAVA*. J. Wiley & Sons, first edition, 1997.
53. J. Grov and P. C. Ölveczky. Increasing consistency in multi-site data stores: Megastore-CGC and its formal analysis. In *Proc. Software Engineering and Formal Methods (SEFM 2014)*, volume 8702 of *Lecture Notes in Computer Science*. Springer, 2014.
54. R. Guerraoui and A. Schiper. Genuine atomic multicast in asynchronous distributed systems. *Theoretical Computer Science*, 254(1-2):297–316, 2001.
55. H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

56. A. Hartmanns and H. Hermanns. The Modest toolset: An integrated environment for quantitative modelling and verification. In *Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2014)*, volume 8413 of *Lecture Notes in Computer Science*. Springer, 2014.
57. J. Hendrix, J. Meseguer, and H. Ohsaki. A sufficient completeness checker for linear ordered-sorted specifications modulo axioms. In *Proc. Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Computer Science*. Springer, 2006.
58. S. Kamin and J.-J. Lévy. Two generalizations of the recursive path ordering. Unpublished Note, Department of Computer Science, University of Illinois, Urbana, IL, 1980.
59. M. Katelman, J. Meseguer, and J. Hou. Redesign of the LMST wireless sensor protocol through formal modeling and statistical model checking. In *Proc. Formal Methods for Open Object-Based Distributed Systems (FMOODS 2008)*, volume 5051 of *Lecture Notes in Computer Science*. Springer, 2008.
60. B. Kirkerud. Lecture notes on rewrite systems. Dept. of Informatics, University of Oslo, 1994. <http://heim.ifi.uio.no/~in307/notater/>.
61. T. Kleinjung et al. Factorization of a 768-bit RSA modulus. In *Proc. Advances in Cryptology (CRYPTO 2010)*, volume 6223 of *Lecture Notes in Computer Science*. Springer, 2010.
62. D. E. Knuth and P. B. Bendix. Simple word problems in universal algebras. In J. Leech, editor, *Computational Problems in Abstract Algebra*, pages 263–297. Pergamon Press, 1970.
63. M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Proc. Computer Aided Verification (CAV 2011)*, volume 6806 of *Lecture Notes in Computer Science*. Springer, 2011.
64. L. Lamport. The part-time parliament. *ACM Transactions on Computer Systems*, 16(2):133–169, 1998.
65. L. Lamport. Paxos made simple. *ACM SIGACT News*, 32:51–58, 2001.
66. B. Lampson and H. Sturgis. Crash recovery in a distributed data storage system. Technical report, Xerox Palo Alto Research Center, 1976.
67. F. Laroussinie, N. Markey, and P. Schnoebelen. Temporal logic with forgettable past. In *Proc. Logic in Computer Science (LICS 2002)*. IEEE Computer Society, 2002.
68. D. Lepri, E. Abraham, and P. C. Ölveczky. Sound and complete timed CTL model checking of timed Kripke structures and real-time rewrite theories. *Science of Computer Programming*, 99:128–192, 2015.
69. E. Lien and P. C. Ölveczky. Formal modeling and analysis of an IETF multicast protocol. In *Proc. Software Engineering and Formal Methods (SEFM 2009)*. IEEE Computer Society, 2009.
70. S. Liu, J. Ganhotra, M. R. Rahman, S. Nguyen, I. Gupta, and J. Meseguer. Quantitative analysis of consistency in NoSQL key-value stores. *Leibniz Transactions on Embedded Systems*, 4(1):03:1–03:26, 2017.
71. S. Liu, M. R. Rahman, S. Skeirik, I. Gupta, and J. Meseguer. Formal modeling and analysis of Cassandra in Maude. In *Proc. Formal Methods and Software Engineering (ICFEM 2014)*, volume 8829 of *Lecture Notes in Computer Science*. Springer, 2014.
72. G. Lowe. An attack on the Needham-Schroeder public-key authentication protocol. *Information Processing Letters*, 56:131–133, 1995.
73. G. Lowe. Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In *Proc. Tools and Algorithms for Construction and Analysis of Systems (TACAS 1996)*, volume 1055 of *Lecture Notes in Computer Science*. Springer, 1996.
74. R. R. Lutz. Analyzing software requirements errors in safety-critical embedded systems. In *Proc. IEEE International Symposium on Requirements Engineering*. IEEE, 1993.
75. Z. Manna and A. Pnueli. *Temporal Verification of Reactive Systems: Safety*. Springer, 1995.
76. N. Marti-Oliet, M. Palomino, and A. Verdejo. Rewriting logic bibliography by topic: 1990–2011. *Journal of Logic and Algebraic Programming*, 81(7-8):782–815, 2012.

77. Y. Matijasevich. Simple examples of undecidable associative calculi. *Soviet Mathematics Doklady*, 8(2):555–557, 1967.
78. S. Meier, B. Schmidt, C. Cremers, and D. A. Basin. The TAMARIN prover for the symbolic analysis of security protocols. In *Proc. Computer Aided Verification (CAV 2013)*, volume 8044 of *Lecture Notes in Computer Science*. Springer, 2013.
79. A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996. <http://www.cacr.math.uwaterloo.ca/hac>.
80. J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96:73–155, 1992.
81. J. Meseguer. A logical theory of concurrent objects and its realization in the Maude language. In *Research Directions in Concurrent Object-Oriented Programming*. MIT Press, 1993.
82. J. Meseguer. Membership algebra as a logical framework for equational specification. In *Proc. Recent Trends in Algebraic Development Techniques (WADT 1997)*, volume 1376 of *Lecture Notes in Computer Science*. Springer, 1998.
83. J. Meseguer. The temporal logic of rewriting: A gentle introduction. In *Concurrency, Graphs and Models*, volume 5065 of *Lecture Notes in Computer Science*. Springer, 2008.
84. J. Meseguer. Twenty years of rewriting logic. *Journal of Logic and Algebraic Programming*, 81(7-8):721–781, 2012.
85. J. Meseguer and J. A. Goguen. Initiality, induction and computability. In *Algebraic Methods in Semantics*, pages 460–541. Cambridge University Press, 1985.
86. J. Meseguer and G. Rosu. The rewriting logic semantics project. *Theoretical Computer Science*, 373(3):213–237, 2007.
87. J. Meseguer and G. Rosu. The rewriting logic semantics project: A progress report. *Information and Computation*, 231:38–69, 2013.
88. R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.
89. P. C. Ölveczky. Semantics, simulation, and formal analysis of modeling languages for embedded systems in Real-Time Maude. In *Formal Modeling: Actors, Open Systems, Biological Systems*, volume 7000 of *Lecture Notes in Computer Science*. Springer, 2011.
90. P. C. Ölveczky. Real-Time Maude and its applications. In *Proc. Rewriting Logic and Its Applications (WRLA'14)*, volume 8663 of *Lecture Notes in Computer Science*. Springer, 2014.
91. P. C. Ölveczky, A. Boronat, and J. Meseguer. Formal semantics and analysis of behavioral AADL models in Real-Time Maude. In *Proc. Formal Techniques for Distributed Systems (FORTE 2010)*, volume 6117 of *Lecture Notes in Computer Science*. Springer, 2010.
92. P. C. Ölveczky and J. Meseguer. Specification of real-time and hybrid systems in rewriting logic. *Theoretical Computer Science*, 285:359–405, 2002.
93. P. C. Ölveczky and J. Meseguer. Semantics and pragmatics of Real-Time Maude. *Higher-Order and Symbolic Computation*, 20(1-2):161–196, 2007.
94. P. C. Ölveczky, J. Meseguer, and C. L. Talcott. Specification and analysis of the AER/NCA active network protocol suite in Real-Time Maude. *Formal Methods in System Design*, 29(3):253–293, 2006.
95. P. C. Ölveczky and S. Thorvaldsen. Formal modeling, performance estimation, and model checking of wireless sensor network algorithms in Real-Time Maude. *Theoretical Computer Science*, 410(2-3):254–280, 2009.
96. L. L. Peterson and B. S. Davie. *Computer Networks: A Systems Approach*. Morgan Kaufmann, second edition, 2000.
97. A. Pnueli. The temporal logic of programs. In *Proc. Foundations of Computer Science (FOCS 1977)*. IEEE Computer Society, 1977.
98. R. L. Rivest, A. Shamir, and L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

99. J. Rushby. Mechanized formal methods: Progress and prospects. In *Proc. Foundations of Software Technology and Theoretical Computer Science (FSTTCS 1996)*, volume 1180 of *Lecture Notes in Computer Science*. Springer, 1996.
100. R. Sasse, S. T. King, J. Meseguer, and S. Tang. IBOS: A correct-by-construction modular browser. In *Proc. Formal Aspects of Component Software (FACS 2012)*, volume 7684 of *Lecture Notes in Computer Science*. Springer, 2012.
101. S. Sebastio and A. Vandin. MultiVeStA: Statistical model checking for discrete event simulators. In *Proc. Performance Evaluation Methodologies and Tools (ValueTools 2013)*. ICST, Brussels, Belgium, 2013.
102. K. Sen, M. Viswanathan, and G. Agha. On statistical model checking of stochastic systems. In *Proc. Computer Aided Verification (CAV 2005)*, volume 3576 of *Lecture Notes in Computer Science*. Springer, 2005.
103. K. Sen, M. Viswanathan, and G. A. Agha. VESTA: A statistical model-checker and analyzer for probabilistic systems. In *Proc. Quantitative Evaluation of Systems (QEST 2005)*. IEEE Computer Society, 2005.
104. P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal of Computing*, 26(5):1484–1509, 1997.
105. Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 2003.
106. Y. Toyama. Counterexamples to termination for the direct sum of term rewriting systems. *Information Processing Letters*, 25:141–143, 1987.
107. M. Y. Vardi. Branching vs. linear time: Final showdown. In *Proc. Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2001)*, volume 2031 of *Lecture Notes in Computer Science*. Springer, 2001.
108. M. Wirsing. Algebraic specification. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B. Elsevier, 1990.
109. H. L. S. Younes and R. G. Simmons. Probabilistic verification of discrete event systems using acceptance sampling. In *Proc. Computer Aided Verification (CAV 2002)*, volume 2404 of *Lecture Notes in Computer Science*. Springer, 2002.

Index

A

algebra, 110
 canonical term algebra, 118, 122
 computable, 24
 ground term algebra, 115
 initial algebra, 120
 isomorphic, 114
 many-sorted, 110
 normal form algebra, 118
 order-sorted, 112
 quotient algebra, 117
 (Σ, E) -algebra, 116
 $\mathcal{T}_{\Sigma, E}$, 117
 term algebra, 115
alternating bit protocol, 205
arity, 16
associativity, 42
atomic commit, 212
atomic multicast, 193
authentication, 1, 233

B

behavior, 142
Bernoulli distribution, 295
binary tree, 26, 106
BINTREE-NAT1, 26
Birkhoff's Completeness Theorem, 119
blackjack, 176
 statistical model checking of, 298
BOOL, 35
BOOLEAN, 14
broadcast, 189
 wireless, 190
built-in module, 35
 Boolean values, 35
 floating-point numbers, 39
 integers, 38

 natural numbers, 36
 random numbers, 40
 rational numbers, 38
 strings, 39

C

canonical form, 63
category, 144
choice operator, 131
class declaration, 164
class inheritance, 165
 multiple inheritance, 166
coffee bean game, 133, 149
comment, 12
communication, 183
 asynchronous, 160, 183
 ordered, 183, 193
 synchronous, 157, 183, 184
 unordered, 184
 unordered and asynchronous, 185
 unreliable, 191
commutativity, 41
computation, 19, 63
computation tree logic (CTL), 282
concurrency, 135
 nested concurrency, 138
 sideways concurrency, 136
CONFIGURATION, 163
configuration, 156
confluence, 63, 85, 90
 ground confluence, 85
 local confluence, 86
congruence, 301
connected component, 30
consensus, 231
 Paxos consensus algorithm, 232
consistency, 212

constant, 16
 constructor, 13, 17
 constructor ground term, 13, 20, 102
 CONVERSION, 40
 critical pair, 89
 critical section, 221
 cryptographic protocol, 1, 233
 CTL*, 282

D

deadlock, 173
 debugging, 58
 declarative program, 11
 definedness, 21
 denotational semantics, 109
 derivation, 19, 63
 digital signature, 234
 dining philosophers problem, 170
 LTL model checking of, 280
 distributed algorithm, 211
 distributed system, 128

E

embedding, 77
 empty sort, 124
 equation, 18
 conditional, 18
 equational attribute, 41
 equational completion, 90
 equational logic, 93, 94
 decidability, 99
 deduction rules, 94
 many-sorted, 124
 soundness and completeness, 118
 undecidability, 96
 equivalence class, 300
 of terms, 41
 equivalence relation, 300
 error sort, 34
 evaluation strategy, 57
 eager evaluation, 57
 lazy evaluation, 57
 event, 251
 event-based property, 250

F

FACTORIAL, 37
 fairness, 173, 255, 271
 compassion, 255
 justice, 255
 FLOAT, 39

football, 132, 250
 formatting, 58
`fresh`, 145, 147, 148
 frozen operator, 144
 Full Maude, 155, 162
 obtaining search path, 169
 search, 168
 transform to core Maude, 169
 function, 299
 composition, 299
 identity, 300
 injective, 299
 inverse, 299
 lambda notation, 299
 partial, 31, 299
 surjective, 299
 function symbol, 16

G

GRAPH, 52
 ground term, 16
 group, 91, 93, 100, 112

H

Hilbert's Tenth Problem, 102
 homomorphism, 112

I

identity element, 43
 inductive theorem, 94, 101, 102
 associativity of addition, 104
 commutativity of addition, 105
 induction scheme, 105
 lemma, 104
 INT, 38
 integers, 31
 intended model, 109, 120
 interleaving semantics, 128
 intruder, 241
 Dolev-Yao model, 241
 isomorphism, 114

J

joinable, 89

K

kind, 34
 Knuth-Bendix completion, 100
 Kripke structure, 268
 Kruskal's Theorem, 77

L

label, 130
 language semantics, 6
 leader election, 226
 ring-based algorithm, 226
 spanning-tree-based algorithm, 227
 least sort, 30
 lexicographic comparison, 28, 76
 lexicographic path order (lpo), 79
 implementation, 83
 linear temporal logic (LTL), 263
 formula, 265
 model checking in Maude, 273
 satisfiability and tautology checking, 281
 semantics, 267
 LINK, 194
 link, 193
 limited capacity, 196
 unreliable, 195
 list, 24, 43
 LIST-INT, 44
 LIST-NAT1, 25
 livelock, 173
 looping, 72

M

many-sorted equational specification, 12, 18
 expressiveness, 23
 matching, 61
 modulo axioms, 65
 matching equation, 147
 mathematical induction, 301
 Maude, 4
 applications, 5
 comments, 12
 download, 5, 13
 errors, 14
 functional module, 12
 module importation, 15
 run, 13
 system module, 131
 Windows, 5
 membership equational logic, 34
 mergesort, 48
 parametric, 55
 message delay, 290
 message passing, 159
 message wrapper, 188

MESSAGE-LOSS, 191
 MESSAGE-LOSS-DUPLICATION, 191
 MESSAGE-WRAPPER, 188
 metric temporal logic, 292
 monotonic, 74
 MSET-INT, 44
 MULTICAST, 189
 multicast, 188
 multiset, 44
 multiset comparison, 76
 multiset path order (mpo), 80
 mutual exclusion, 221
 central server algorithm, 223
 Maekawa's voting algorithm, 222, 225
 temporal logic model checking of, 277
 token ring algorithm, 222, 225

N

NAT, 37
 NAT-ADD, 13
 NAT-EXP, 24
 NAT-MULT, 23
 NAT<, 15
 natural numbers, 13
 Needham-Schroeder protocol (NSPK), 1,
 233, 235
 attack on, 245
 Lowe's correction, 248
 Newman's Lemma, 86
 no confusion, 123
 no junk, 123
 nonce, 235
 nondeterminism, 128
 nontermination, 72
 Toyama's example, 73
 normal form, 21, 63
 NP-complete problem, 49
 Clique, 50
 Hamiltonian Circuit, 50, 52
 Integer Knapsack, 54
 Knapsack, 50
 Multiprocessor Scheduling, 50
 Partition, 50
 Satisfiability, 49
 Subgraph Isomorphism, 50
 Subset Sum, 50, 51, 57
 Traveling Salesman, 50, 54, 134
 NSPK, 237

O

object, 155
 creation and deletion, 158
 identifier, 164
 object-oriented module, 163
 ONE-PERSON, 133
 one-sorted, 59
 one-step concurrent rewrite, 141
 OO-POPULATION, 160
 operational semantics, 18, 59
 operator, 16
 operator attribute
 assoc, 42
 comm, 41
 ctor, 13
 ditto, 37
 format, 58
 frozen, 144
 id:, 43
 prec, 15
 special, 36
 strat, 57
 optimal proof system, 102
 order-sorted specification, 29
 overloaded, 29
 owise, 57

P

PARAM-SORT, 55
 parameterized module, 54
 partial order, 300
 past temporal operator, 282
 POPULATION, 137, 165
 position, 60
 powerset, 299
 precedence, 15, 79
 prelude.maude, 35
 preregular, 30
 probabilistic rewrite theory, 294
 to ordinary rewrite theory, 295
 probabilistic system, 293
 probabilistic temporal logic, 296
 process failure, 219
 Byzantine failure, 219
 crash failure, 218
 fault injection, 220
 recovery, 219
 protocol, 188
 public-key cryptography, 233, 234
 RSA algorithm, 234

Q

quicksort, 47

R

RANDOM, 40
 random numbers, 40
 RAT, 38
 reactive system, 127
 Real-Time Maude, 292
 real-time system, 283
 in rewriting logic, 284
 reduces, 59
 reducible, 63
 reduction, 59
 reduction sequence, 19, 63
 reduction step, 62
 relation, 300
 renaming, 87
 replicated databases, 212
 requirement specification, 249
 rew, 145, 147, 148
 rewrite condition, 131
 rewrite rule, 130
 rewrite theory, 131
 rewriting, 59
 rewriting logic, 127, 130
 concurrent steps, 140
 confluence, 142
 deduction rules, 140
 execution, 145
 model, 144
 semantics, 144
 sequent, 130
 specification, 131
 termination, 142
 run, 142

S

search, 150
 show path, 152
 search, 145, 150, 168
 self-embedding, 77
 separation problem, 185
 SEQNO-UNORDERED, 201
 sequence number, 200
 sequent, 94, 139
 sequential rewrite, 141
 shared variable, 184, 197
 signature
 many-sorted, 16
 order-sorted, 29

simplification, 59
simplification order, 76, 78, 81
simplification step, 62
simulation, 147
 randomized, 176, 293
sliding window protocol, 206
 with links, 209
sort, 12, 16
spacecraft, 4
starvation, 173
state proposition, 252
state-based property, 251
statistical model checking, 293
 PVESTA, 297
 QUATEX properties, 297
strict partial order, 75, 300
 well-founded, 75
STRING, 39
strings, 39
subclass, 165
subsort, 29
 semantic, 33
substitution, 61
subterm, 60
 proper, 61
sufficient completeness, 22
symmetric-key cryptography, 235

T

temporal logic of rewriting, 281
temporal properties, 252
 guarantee, 254
 invariance, 253
 analysis, 260
 reachability, 255
 response/reactivity, 256
 stability, 256

 until, 257
termination, 20, 67
 operational, 65
 undecidability, 68
 weakly terminating, 67
 weight function, 74, 81
terms, 17
tick rewrite rule, 284
TOTAL-ORDER, 55
tracing, 19, 58, 148
transaction, 211
 distributed, 211
transport protocol, 199
Turing machine, 68, 135
 configuration, 69
 universal Turing machine, 72
two-phase commit protocol (2PC), 211, 212
 with process failures, 220

U

unicast, 185
unification algorithm, 87
unifier, 87
 most general, 87
uniform distribution, 295
unsorted, 59

V

variable, 13, 17
 declaration, 17
variable assignment, 116
variable substitution, 61
view, 55

W

web browser attack, 5
whiteboard game, 134