The Associative Structure of State Machines

An Associative Algebra Approach to Logic, Arithmetic and Automata

NICO F. BENSCHOP
Preface

This book is intended for researchers at industrial laboratories, teachers and students at technical universities, in electrical engineering, computer science and applied mathematics departments, interested in new developments of modeling and designing digital networks (DN: state machines, sequential and combinational logic) in general, as a combined math/engineering discipline. As background an undergraduate level of modern applied algebra\(^1\) will suffice. Essential concepts and their engineering interpretation are introduced in a practical fashion with examples. The motivation in essence is: the importance of the unifying associative algebra of function composition (semigroup theory) for the practical characterisation of the three main functions in computers, namely sequential logic (state-machines), arithmetic and combinational (Boolean) logic.

Known principles of discrete mathematics, especially finite semigroups, residue arithmetic and boolean logic (lattices) are interpreted in terms of practical DN design issues. The main three levels of state machine synthesis form a natural 'top down' hierarchy of associative algebras:

<table>
<thead>
<tr>
<th>Application</th>
<th>Algebra type</th>
<th>Syntax</th>
<th>Objects</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>sequential logic</td>
<td>associative</td>
<td>((ab)c = a(bc))</td>
<td>functions</td>
<td>sequence</td>
</tr>
<tr>
<td>arithmetic</td>
<td>commutative</td>
<td>(ab = ba)</td>
<td>numbers</td>
<td>(+) (-)</td>
</tr>
<tr>
<td>combinat'l logic</td>
<td>idempotent</td>
<td>(aa = a)</td>
<td>sets</td>
<td>(\cup) (\cap)</td>
</tr>
</tbody>
</table>

Historically, non-commutative and idempotent algebras diverged from arithmetic in the nineteenth century. Our aim is to emphasize again their arithmetic nature, for practical engineering purposes such as efficient synthesis of binary logic and state machines. The 'static' (combinational, idempotent, \(x^2 \equiv x\)) and 'iterative' (commutative, \(x^{i+1} = x^i x = x x^i\)) aspects can be modeled by finite residue arithmetic. Apart from the two non-commutative components of memory type (branch- and reset- machines, shown to be each others dual), non-commutative aspects of sequential behaviour can be represented by coupling functions between components.

- In the first of three parts, on state machines (Ch.1–4), an introductory chapter recalls basic principles in theory and practice. The five basic components of

---

\(^1\)Birkhoff-Bartee 1970 - *Modern Applied Algebra*

Hartmanis-Stearns 1970 - *Algebraic Structure of Sequential Machines*
sequential behaviour (with indecomposable semigroup) are derived, with ways to couple them efficiently - only required in the non-commutative case. They define the five basic types of state machines for network composition.

• In the second part, on combinational (Boolean) logic (Ch.5,6) the concept of spectrum as a characteristic sequence of numbers, is borrowed from Fourier analysis for order-independent (symmetric) synthesis of Boolean functions (BFs).

A useful arithmetic compositional rule holds: the spectrum of a product of functions (of disjoint inputs) is the product of the component spectra. In fact Boole (1854) introduced his algebra - a calculus of binary properties - as an idempotent form of arithmetic. This allows convolution-like composition rules (as in linear filters), to be developed.

Symmetric BFs are implemented as a crossing-free and compact orthogonal grid network of MOS transistors in the silicon plane, to obtain a regularly structured VLSI implementation. Simply removing transistors from such grid yields planar BFs with the desired crossing-free property, covering a majority of Boolean functions. Using this representation, the complexity of BFs grows polynomial, and not exponential, with the number of inputs. It appears that by permuting and/or inverting the $n$ inputs, each $BF_n$ for $n \leq 4$ is planar. A fast $O(n^2)$ algorithm for symmetric logic synthesis is developed, and applied to optimize fault-tolerant logic using Hamming- or product- codes for error correction, with synthesized gate count as cost criterium.

• The third and last part, on arithmetic (Ch.7-11), analyses residue arithmetic with two extremal types of prime related moduli: $p^k$ and $m_k = p_1 p_2 \ldots p_k$ typical for 'sequential' resp. 'parallel' arithmetic. By expanding $r \mod m$ residues with a 'carry' $c$ as multiple of modulus $m$: $n = cm + r$, integer arithmetic obtains a dual focus on closure- and generative properties of residues and carry, as independent resp. dependent network components. This balanced approach to arithmetic provides new insights into old and well known problems in finite additive number theory (Fermat, Goldbach, Waring: Ch.8,9,10) with practical engineering results. For instance each odd residue mod $2^k$ is a unique signed power of 3, allowing efficient log-arithmetic over bases 2 and 3 [patent US–5923888]. Moreover, a binary log-arithmetic microprocessor (32 bits, in 0.18 $\mu$ CMOS technology) is described, designed as part of a European Esprit project\(^2\), comparing favourably with floating point arithmetic devices.

\[ Nim P. Benschop \quad \diamond \quad \text{Amspade Research, Geldrop, Netherlands, 2010.} \]

\(^2\)Esprit 33544 HSLA, 1999-2002, main contractor Univ.Newcastle (dpt.ECE) UK
# Introduction

1 Sequential and combinational Logic ............................................. 2  
1.1 Five basic state machines: network components ...................... 5  
1.3 Subset/partition, local/global, additive/multi’ve ....................... 8  
1.3.1 Associative closure: semigroup and sub-semigroup ........... 9  
1.3.2 Preserved partition: congruence and image ...................... 10  
1.4 Integer arithmetic: residues with carry ............................ 12

# Simple Semigroups and the Five Basic Machines

2.1 State Machine: Sequential Closure and Rank .......................... 14  
2.2 Basic Machines and Simple Semigroups .............................. 16  
2.2.1 Iterations: monotone, periodic, idempotent ................ 17  
2.2.2 Ordered idempotents $H$ for combinational logic .......... 18  
2.2.3 The five minimal semigroups and basic machines ......... 19  
2.3 Equivalent idempotents: memory components $L, R$ ............. 21  
2.4 Maximal Subgroups: periodic $G$ .................................. 26  
2.5 Constant Rank Machines, simple semigroups .......................... 28

# Coupling State Machines

3.1 Introduction ..................................................................... 32  
3.2 No coupling: semigroup $Z(.) \bmod m$, any $m$ ........................ 33  
3.3 Decompose machine: right congruence suffices .................. 39  
3.4 Cascade composition: full groups $FG_3$ and $FG_4$ ............ 42  
3.5 Decomposing the full- and alternating group over four states .. 47  
3.6 Decompose simple group $AG_n \subset FG_n$ for $n > 4$ .......... 50  
3.7 Loop composition superfluous .................................. 55

# General Decomposition of State Machines

4. General Decomposition of State Machines ................................. 59
4.1 Introduction ................................................. 59
4.2 Implementing $M = (Q,A)$ by its alphabet $A$ ............. 60
   4.2.1 Decomposition by local input closures ............... 61
4.3 Bottom-up decomposition of $S = A^* / Q$ .................. 62
4.4 Partial direct products, unused codes, efficiency .......... 63
4.5 Example .................................................. 63
   4.5.1 Top-down decomposition by local input closures .... 66
   4.5.2 Global decomposition by maximal iterative components 67
4.6 Invariants: ordered commuting idempotents ............... 69

5 Symmetric and Planar Boolean Logic Synthesis .......... 79
   5.1 Introduction ............................................. 80
   5.2 Logic Synthesis independent of input ordering .......... 81
      5.2.1 Orthogrid plot and rank spectrum ................. 81
      5.2.2 Factoring paths by a planar node ................ 82
   5.3 Symmetric and Threshold $BF$'s ....................... 84
      5.3.1 Symmetric functions 'count' ..................... 84
      5.3.2 $T$-cell library, threshold logic cells .......... 85
   5.4 Planar cut and factoring ................................ 86
   5.5 Fast symmetric synthesis: quadratic in nr. inputs .... 87
   5.6 Experiments and conclusion ............................. 88
   5.7 Planar Boolean logic synthesis .......................... 89
      5.7.1 All $BF_n$ are planar upto $n=4$ inputs .......... 90

6 Fault Tolerant Logic with Error Correcting Codes .......... 97
   6.1 Introduction ............................................. 98
   6.2 Fault tolerant IC design environment .................... 99
      6.2.1 Implementation at register transfer level ......... 99
      6.2.2 Protecting registers and connections ............. 101
6.3 Three logic circuit error correction methods .................................. 101
  6.3.1 Majority voting ................................................................. 102
  6.3.2 Hamming codes (block codes) ........................................... 103
  6.3.3 Product codes (array codes) ............................................. 103

6.4 Demonstration of experimental circuit ........................................... 104
6.5 Results for typical designs ........................................................ 110
6.6 Conclusions .................................................................................. 113

7 Fermat’s Small Theorem extended to $r^{p-1} \mod p^3$ ............................ 116
  7.1 Introduction .................................................................................. 116
    7.1.1 Divisors $r \mid p \pm 1$ and residues $(p \pm 1)^n \mod p^3$ ....... 118
  7.2 Lattice structure of semigroup $\mathbb{Z}(.) \mod q$ ............................ 119
    7.2.1 Distinct $e^{p-1} \mod p^3$ for idempotents $e \in \mathbb{Z}_{p-1}$ .... 120
  7.3 Distinct $r^{p-1} \mod p^3$ for divisors $r \mid p \pm 1$ ......................... 123
    7.3.1 Idempotents of $\mathbb{Z}_{p+1}(.)$ and divisors of $p + 1$ ......... 123

8 Additive structure of units group $\mod p^k$, Fermat’s Last Theorem
by carry extension ............................................................................. 126
  8.1 Introduction .................................................................................. 127
  8.2 Structure of the group $G_k$ of units ........................................... 129
  8.3 Cubic root solution in core, and core symmetries ......................... 131
    8.3.1 Another derivation of the cubic roots of 1 $\mod p^k$ ............ 133
    8.3.2 Core increment symmetry $\mod p^{2k+1}$, asymmetry $\mod p^{3k+1}$ ............................. 133
  8.4 Symmetries as functions yield ‘triplets’ ..................................... 135
    8.4.1 A triplet for each unit $n$ in $G_k$ ......................................... 137
    8.4.2 The EDS argument extended to non-core triplets .......... 138
  8.5 Relation to Fermat’s Small and Last Theorem ............................. 140
    8.5.1 Proof of the FLT inequality ................................................ 141
8.6 Conclusions and Remarks ........................................ 142

9 Additive structure of \( \mathbb{Z}(\ ) \mod m_k \) (squarefree) and Goldbach’s conjecture 146
9.1 Introduction .................................................. 147
9.2 Lattice of groups .............................................. 149
  9.2.1 Ordering of commuting idempotents .................... 149
  9.2.2 Lattice of idempotents: add vs join .................... 150
9.3 Primes, composites and neighbours .......................... 151
  9.3.1 Each idempotent’s successor is in \( G_1 \) or \( G_2 \) ...... 152
9.4 Prime units and carry extension .............................. 153
  9.4.1 Pair sums of carry extended units .................... 155
  9.4.2 Pair sums of primes in \( G(3) \) ......................... 155
  9.4.3 Exclude composites, base-primes and 1 as summands 157
9.5 Proving GC by induction, or by reduction and contradiction 158
9.6 Conclusions .................................................. 159

10 Powersums \( x^p \) represent residues mod \( p^k \), from Fermat to Waring 162
10.1 Introduction .................................................. 163
10.2 Core increments as coset generators ........................ 164
10.3 Core extensions \( A_k \) to \( F_k \), pairsums mod \( p^k \) ........ 166
10.4 Conclusions .................................................. 171

11 Log-arithmetic, with single and dual base 173
11.1 Log-arithmetic with dual base 2 and 3 ..................... 173
  11.1.1 Proposed new binary number code .................... 174
  11.1.2 Example ................................................ 175
  11.1.3 Application to multipliers ............................ 176
  11.1.4 Signed magnitude binary code over bases 2 and 3 ... 176
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1.5</td>
<td>Addition in log code: 'odd' arithmetic (base 2 and 3)</td>
<td>177</td>
</tr>
<tr>
<td>11.2</td>
<td>European Logarithmic Microprocessor <em>ELM</em></td>
<td>179</td>
</tr>
<tr>
<td>11.2.1</td>
<td>Introduction: Log-arithmetic with single base 2</td>
<td>180</td>
</tr>
<tr>
<td>11.2.2</td>
<td>Log-arithmetic algorithms, an overview</td>
<td>182</td>
</tr>
<tr>
<td>11.2.3</td>
<td>Data format, range and precision</td>
<td>185</td>
</tr>
<tr>
<td>11.2.4</td>
<td>Measurement of Accuracy</td>
<td>185</td>
</tr>
<tr>
<td>11.2.5</td>
<td>Conventional <em>LNS</em> Addition and Subtraction</td>
<td>187</td>
</tr>
<tr>
<td>11.2.6</td>
<td>New Error Correction Algorithm</td>
<td>188</td>
</tr>
<tr>
<td>11.2.7</td>
<td>Error Correction for Subtraction</td>
<td>193</td>
</tr>
<tr>
<td>11.2.8</td>
<td>Adder/Subtractor design and evaluation</td>
<td>193</td>
</tr>
<tr>
<td>11.2.9</td>
<td>Architecture and performance</td>
<td>195</td>
</tr>
<tr>
<td>11.2.10</td>
<td>VLSI Implementation</td>
<td>197</td>
</tr>
<tr>
<td>11.2.11</td>
<td>The <em>ELM</em>: some more architectural details</td>
<td>198</td>
</tr>
<tr>
<td>11.2.12</td>
<td>Accuracy comparisons <em>LNS</em> vs. <em>FLP</em></td>
<td>200</td>
</tr>
<tr>
<td>11.2.13</td>
<td>The TMS-320C6711</td>
<td>201</td>
</tr>
<tr>
<td>11.2.14</td>
<td>Conclusion</td>
<td>203</td>
</tr>
</tbody>
</table>

- **Bibliography** 207
- **Index** 213
Author cv

1962 - 1966 MSc. EE / TU-Delft (The Netherlands)
1967 - 1970 PhD. EE / Waterloo Univ. (Ont., Canada)
   Research in Digital IC design methods.

Review 1


As a historical fact, mathematics developed from applications – in rational mechanics and number theory – for which commutative algebra was most natural. If the basic applications were from network theory (Turing machines) the associative algebra \((ab)c = a(bc)\) would have been more natural, with Boolean algebra \(aa = a\) and commutative algebra \(ab = ba\) as special cases.

Benschop develops this thesis in an idiosyncratic fashion, reinforced by a long career of practical experience. This book may well be an important historical document, also useful for seminars, even if it is not presented primarily for class usage.

There are profuse illustrations in classic number theory, as well as claims that the outlook sheds new light on classic problems such as those of Fermat and Goldbach, interpreted as machines. As unlikely as it is that this may be practical, it makes for an interesting book.

[Fermat, Ch.8] http://pc2.iam.fmph.uniba.sk/amuc/_vol74n2.html (p169-184)

Review 2

Zentralblatt MATH, Vol.1169, 2009
(c 2010 FIZ Karlsruhe io-port 05500994)


Keywords: boolean functions; state machines; sequential logic; combinatorial logic.

The book presents new ways for modeling digital networks (state machines, sequential and combinational logic). It contains applications for known principles of discrete mathematics.

The book has three parts. The first part presents state machines and some algebraic ways to model them. Basically, network composition is reduced to five basic types of state machines. The second part is about Boolean logic. It introduces the concept of spectrum and some applications and algorithms that are using it. It also presents symmetric Boolean functions and some of their properties. An algorithm for symmetric logic synthesis is provided. The last part of the book is about residue arithmetic with two extremal types of prime related moduli. A focus on closure- and generative properties of residues and carry is obtained.

The book also presents new ideas on finite additive number theory and a binary log-arithmetic microprocessor. This book can be very useful for students and professors and also for researchers interested in digital network theory. It covers a lot of fields, ranging from electrical engineering to computer science and applied mathematics.

prof. Eleonor Ciurea (Univ. Brasov, Romania)
doi:10.1007/978-1-4020-9865-9