

W.Kluge

Abstract Computing Machines

A Lambda Calculus Perspective

With 89 Figures

Contents

1	Introduction	1
2	Algorithms and Programs	11
2.1	Simple Algorithms	14
2.1.1	Getting Started with Some Basics	15
2.1.2	Recursive Functions	19
2.1.3	The Termination Problem	23
2.1.4	Symbolic Computations	24
2.1.5	Operating on Lists	30
2.2	A Word on Typing	31
2.3	Summary	34
3	An Algorithmic Language	37
3.1	The Syntax of AL Expressions	38
3.2	The Evaluation of AL Expressions	41
3.3	Summary	47
4	The A-Calculus	51
4.1	A-Calculus Notation	52
4.2	λ -Reduction and α -Conversion	53
4.3	An Indexing Scheme for Bound Variables **	59
4.4	The Nameless λ -Calculus	63
4.5	Reduction Sequences	68
4.6	Recursion in the A-Calculus	73
4.7	A Brief Outline of an Applied A-Calculus	78
4.8	Overview of a Typed A-Calculus	79
4.8.1	Monomorphic Types	81
4.8.2	Polymorphic Types	83
4.9	Summary	86

5	The SE(M)CD Machine and Others	89
5.1	An Outline of the Original SECD Machine.	89
5.2	The SE(M)CD Machine	93
5.2.1	The Traversal Mechanism.	94
5.2.2	Doing /J-Reductions.	96
5.2.3	Reducing a Simple Expression	98
5.3	The #SE(M)CD Machine for the Nameless /t-Calculus.	101
5.4	Implementing $\wedge$ -Reductions.	102
5.5	Other Weakly Normalizing Abstract Machines.	105
5.5.1	The $\wedge$ -Machine.	105
5.5.2	The Categorical Abstract Machine.	107
5.6	Summary.	108
6	Toward Full-Fledged A-Calculus Machines	113
6.1	Berklings's String Reduction Machine.	115
6.2	Wadsworth's Graph Reduction Techniques.	121
6.3	The Acr-Calculus Abstract Machine.	125
6.3.1	The Acr-Calculus **.	126
6.3.2	The Abstract Machine **.	130
6.4	Head-Order Reduction.	132
6.4.1	Head Forms and Head-Order /3-Reductions.	134
6.4.2	An Abstract Head-Order Reduction (HOR) Machine **	141
6.5	Summary.	145
7	Interpreted Head-Order Graph Reduction	149
7.1	Graph Representation and Graph Reduction	150
7.2	Continuing with Reductions in the Head.	156
7.3	Reducing the Tails.	160
7.4	An Outline of the Formal Specification of G_{HOR} .	163
7.5	Garbage Collection.	164
7.6	Summary.	167
8	The B-Machine	171
8.1	The Operating Principles of the 5-Machine.	173
8.2	The Instruction Set	174
8.2.1	Instruction Interpretation Without Sharing **.	176
8.2.2	Interpretation Under Sharing in the Head **.	179
8.3	Executing i?-Machine Code: an Example **.	181
8.4	Supporting Primitive Functions.	186
8.5	Summary.	189
9	The G-Machine	193
9.1	Basic Language Issues.	195
9.2	Basic Operating Principles of the G-Machine.	197
9.3	Compiling Supercombinators to G-Machine Code.	201

9.4	G-Code for Primitive Functions.	204
9.5	The Controlling Instructions *.	205
9.6	Some G-Code Optimizations.	209
9.7	Summary.	211
10	The TT-RED Machinery.	215
10.1	The Basic Program Execution Cycle.	215
10.2	The Operating Principles of the Abstract Machines.	221
10.3	The Lazy Abstract Stack Machine LASM.	223
10.3.1	The LASM Instruction Set.	225
10.3.2	Compilation to LASM Code *.	228
10.3.3	Some Simple Code Optimizations.	232
10.4	The Strict Abstract Stack Machine SASM.	235
10.4.1	The SASM Instruction Set.	236
10.4.2	Compilation to SASM Code *.	237
10.4.3	Code Execution.	240
10.5	Reducing to Full Normal Forms *.	244
10.6	Summary.	249
11	Pattern Matching.	253
11.1	Pattern Matching in AL.	253
11.2	Programming with Pattern Matches.	255
11.3	Preprocessing Pattern Matches.	258
11.4	The Pattern Matching Machinery.	260
11.5	Compiling Pattern Matches to LASM Code *.	263
11.6	Code Generation and Execution: an Example **.	265
11.7	Summary.	268
12	Another Functional Abstract Machine.	271
12.1	The Machine and How It Basically Works.	272
12.1.1	Some Semantic Issues.	273
12.1.2	Index Tuples and the Runtime Environment.	275
12.2	The SECD _i Instruction Set.	279
12.3	Compilation to SECD _i Code.	282
12.4	Summary.	285
13	Imperative Abstract Machines.	289
13.1	Outline of an Imperative Kernel Language.	291
13.2	An Example of an IL Program.	294
13.3	The Runtime Environment.	296
13.3.1	Using Static and Dynamic Links.	298
13.3.2	Dropping Dynamic Links.	300
13.3.3	Calculating Stack Addresses *.	302
13.4	The Instruction Set.	304
13.5	Compiling IL Programs to IAM Code *.	306

13.6	Compiling the Bubble-Sort Program	309
13.7	Outline of a Machine for a 'Flat' Language	312
13.8	Summary	318
14	Real Computing Machines	321
14.1	A Typical CISC Architecture	323
14.1.1	The Register Set, Formats and Addressing in Memory	324
14.1.2	Addressing Modes	326
14.1.3	Some Important Instructions	328
14.1.4	Implementing Procedure Calls	330
14.2	A Typical RISC Architecture	334
14.2.1	The SPARC Register Set	335
14.2.2	Some Important SPARC Instructions	339
14.2.3	The SPARC Assembler Code for Factorial	341
14.3	Summary	344
A	Input/Output	347
A.1	Functions as Input/Output Mappings	348
A.2	Continuation-Style Input/Output	354
A.3	Interactions with a File System	357
B	On Theorem Proving	361
	References	369
	Index	377