

# Contents

<b>1</b>	<b>Introduction</b>	1
1.1	Static Average-Case Analysis	2
1.1.1	The Need for Static Average-Case Analysis Tools	2
1.1.2	Compositionality: the Golden Key to Static Analysis	2
1.1.3	The Main Bottleneck for Static Average-Case Analysis	3
1.2	Removing the Bottleneck for Static Average-Case Analysis	4
1.2.1	The Need for Novel Language Design	4
1.2.2	Efficiency-Oriented Languages	4
1.2.3	The Meaning of Static Timing in our Context	5
1.3	The $\mathcal{MOQA}$ Language	6
1.3.1	General Description	6
1.4	Tracking Distributions	12
1.4.1	The Uniform Distribution	12
1.4.2	S-Distributions	13
1.5	Random Bag Preservation	14
1.6	The Necessity of Guaranteeing Random Bag Preservation	17
1.7	A Sufficient Condition for Random Bag Preservation	20
1.7.1	<i>Split</i> : an Illustration of Random Bag Preservation	21
1.7.2	<i>Split</i> : the General Case	26
1.7.3	Tracking S-Distributions in $\mathcal{MOQA}$	26
1.8	$\mathcal{MOQA}$ Operations	27
1.8.1	An Overview of the Basic $\mathcal{MOQA}$ Operations	27
1.8.2	Conditionals, Loops, Recursion	28
1.9	Compositionality	28
1.9.1	Average-Case Time is IO-Compositional	29
1.9.2	Linear-Compositionality Theorem	29
1.10	Related work and advantages of $\mathcal{MOQA}$	30
1.11	Related Areas	32
1.11.1	Bridging Semantics and Complexity	33
1.11.2	Implications for Real-Time Languages	36

<b>2</b>	<b>Introductory Notions</b>	39
2.1	Partial Orders & Hasse Diagrams	40
2.2	Series-Parallel Orders	41
2.3	Trees & Heaps	43
2.4	Basic Sorting Algorithms	44
2.5	Uniform Distribution and Bags	47
2.6	Timing Measures	49
<b>3</b>	<b>Compositionality</b>	51
3.1	Compositionality as a Key to Software Timing	51
3.2	IO-Compositionality	52
3.3	Strict Semi IO-Compositionality for Worst-Case and Best-Case Time	54
3.4	Average-Case Time <i>is</i> IO-Compositional	56
3.5	From IO-Compositionality to Linear-Compositionality	57
<b>4</b>	<b>Random Bag Preservation and Isolated Subsets</b>	65
4.1	Random Structures	65
4.2	Floor and Ceiling Functions	70
4.3	Free Sets of Labels	71
4.4	Free Swaps on Random Structures	73
4.5	Random Bag Preserving Functions	74
4.6	Isolated Subsets	79
4.6.1	Strictly Isolated Subsets	82
4.6.2	Simplified Definitions for SP-Orders	90
4.6.3	Extension Theorem	90
<b>5</b>	<b>Basic <math>\mathcal{MOQA}</math> Operations</b>	95
5.1	The Fundamental Data Structuring Problem	96
5.2	The Random Product	96
5.2.1	The Product of two Finite Partial Orders	97
5.2.2	The Product of Two Data-Labelings	98
5.2.3	The Binary Random Product	106
5.2.4	The Unary Random Product	106
5.3	Random Deletion and Percolation	108
5.3.1	Deleting an Extremal Label	108
5.3.2	Percolation and Deletion of Arbitrary Labels	110
5.4	The Random Projection	118
5.5	The Random Split	120
5.5.1	The Random Split of a Discrete Partial Order	120
5.5.2	Random Split of a Random Structure	121
5.6	Top and Bot Operations	125
5.7	Contractive Operations Revisited	127
5.8	Uniformly RB-preserving Functions Revisited	127
5.9	$\mathcal{MOQA}$ -Constructible Random Bags	128
5.10	$\mathcal{MOQA}$ -Constructible Random Bags are Series-Parallel	129

5.11	Simplifications for SP-Orders	129
5.12	Partitions and separative functions	130
<b>6</b>	<b>Average-Case Time of Basic <math>\mathcal{MOQA}</math> Operations</b>	
	<b>Joint with D. Early</b>	133
6.1	Definitions	133
6.2	Average-Case Time	134
6.2.1	The While Condition	134
6.2.2	Push-Up and Push-Down	135
6.3	Series-Parallel Partial Orders	137
6.3.1	Series-Parallel Composition Laws for the $\tau$ Function	137
6.3.2	Series-Parallel Composition Laws for Delete	141
6.4	Examples	144
6.4.1	Calculating the $\tau$ Function	144
6.4.2	Inductively Defined Structures	145
6.4.3	Combinatorial Identities	147
<b>7</b>	<b>The <math>\mathcal{MOQA}</math> Language</b>	149
7.1	Conventions	150
7.2	Variables	151
7.3	Types	151
7.4	Arithmetical Expressions	153
7.5	Boolean Expressions	154
7.6	Boolean Statements	157
7.6.1	Probabilities of Boolean Statements	157
7.6.2	Computing Probabilities of Boolean Statements	159
7.6.3	Reduction to Prime DNF's	160
7.6.4	Probabilities for Prime Conjunctions	161
7.7	Random Structure Expressions	164
7.8	Random Conditional Statements	165
7.9	Recursion	167
7.10	$\mathcal{MOQA}$ Programs	171
7.11	Randomness Preservation	172
7.12	Compositional Determination of Average-Case Time	173
7.13	Linear-Compositionality for $\mathcal{MOQA}$ Programs	174
<b>8</b>	<b>Examples of <math>\mathcal{MOQA}</math> Programs</b>	179
8.1	Insertionsort	179
8.2	Merge	179
8.3	Mergesort	180
8.4	Quicksort	180
8.5	Percolating Heapsort	182
8.5.1	Historical Background	182
8.5.2	Pseudo-Code for Percolating Heapsort	184
8.6	Treap-gen	185

8.6.1	Oriented Binary Trees	185
8.6.2	Treaps in $\mathcal{MOQA}$	186
8.6.3	Treap-Generation	189
8.7	TreapSort	190
8.8	Quickselect	190
<b>9</b>	<b>Average-Case Analysis of <math>\mathcal{MOQA}</math> programs</b>	<b>193</b>
9.1	Insertionsort	193
9.2	Mergesort	195
9.3	Quicksort	195
9.4	Percolating Heapsort	197
9.5	Treap-gen	200
9.6	TreapSort	201
9.7	Quickselect	206
<b>10</b>	<b>Distri-Track</b>	
	<b>Joint with D. Hickey and M. Boubekeur</b>	<b>209</b>
10.1	Analysable Code	209
10.2	<i>Distri-Track</i> Architecture	211
10.2.1	Pre-Analysis	211
10.2.2	The Analyser	213
10.3	Random Bag Trackers	215
10.3.1	Condensed Representations	215
10.3.2	Collective Representations	216
10.4	Calculating the ACET	218
10.5	Preliminary Evaluation Study	219
10.5.1	Real-Time $\mathcal{MOQA}$	219
10.5.2	Evaluation Study Description	220
<b>11</b>	<b>Conclusion and Future Work</b>	<b>225</b>
<b>A</b>	<b>Appendix: Proof of the State Theorem</b>	<b>229</b>
A.1	Depth-Levels	229
A.2	Canonical State	230
A.3	Canonical State Algorithm	234
	<b>References</b>	<b>237</b>
	<b>Index</b>	<b>243</b>

A Modular Calculus for the Average Cost of Data  
Structuring

Schellekens, M.

2008, XXIV, 245 p., Hardcover

ISBN: 978-0-387-73383-8