

Introduction to Choreographies

In concurrent and distributed systems, processes can complete tasks together by playing their parts in a joint plan. The plan, or protocol, can be written as a choreography: a formal description of the overall behaviour that processes should collaborate to implement, like authenticating a user or purchasing an item online. Formality brings clarity, but not only that. Choreographies can contribute to important safety and liveness properties.

This book is an ideal introduction to theory of choreographies for students, researchers, and professionals in computer science and applied mathematics. It covers languages for writing choreographies and their semantics, and principles for implementing choreographies correctly. The text treats the study of choreographies as a discipline in its own right, following a systematic approach that starts from simple foundations and proceeds to more advanced features in incremental steps. Each chapter includes examples and exercises aimed at helping with understanding the theory and its relation to practice.

FABRIZIO MONTESI is Professor of Computer Science at the University of Southern Denmark. He is a Villum Young Investigator and recipient of several awards for science and innovation, including the EAPLS Best PhD Dissertation Award and the Best Thesis in ICT Award from the General Confederation of Italian Industry.

Introduction to Choreographies

FABRIZIO MONTESI

University of Southern Denmark





Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre, New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment, a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of education, learning and research at the highest international levels of excellence.

www.cambridge.org

Information on this title: www.cambridge.org/9781108833769

DOI: 10.1017/9781108981491

© Fabrizio Montesi 2023

This publication is in copyright. Subject to statutory exception and to the provisions of relevant collective licensing agreements, no reproduction of any part may take place without the written permission of Cambridge University Press & Assessment.

First published 2023

A catalogue record for this publication is available from the British Library.

A Cataloging-in-Publication data record for this book is available from the Library of Congress.

ISBN 978-1-108-83376-9 Hardback

Cambridge University Press & Assessment has no responsibility for the persistence or accuracy of URLs for external or third-party internet websites referred to in this publication and does not guarantee that any content on such websites is, or will remain, accurate or appropriate.

Contents

<i>List of Illustrations</i>	<i>page</i> ix
<i>List of Notations</i>	xi
Introduction: Alice, Bob, Concurrency, and Distribution	1
This Book	8
Part I Foundations	11
<i>Introduction to Part I</i>	12
1 Inference Systems	15
1.1 Example: Modelling Flight Connections with Inference Rules	15
1.1.1 Expressing a Connected Graph with Axioms	15
1.1.2 Schematic Variables	16
1.2 Derivations	18
1.2.1 Combining Rules	18
1.2.2 Derivations As Trees	19
1.2.3 Searching for Derivations	20
1.2.4 Formal Definition	22
1.2.5 Subderivations	23
1.2.6 Derivations and Induction	25
1.3 Underivable Propositions	29
1.4 Admissible and Derivable Rules	32
1.4.1 Derivable Rules	33
1.4.2 Admissible Rules	35
1.4.3 Taking Stock	38
2 Simple Choreographies	39
2.1 Syntax	39
2.2 Semantics	40
2.2.1 Communication	42
2.2.2 Out-of-Order Execution	42
2.3 Conventions, Notation, and Terminology for Labelled Transition Systems	45
2.3.1 Conventions for the Definition of Labelled Transition Systems	45
2.3.2 Notation and Terminology for Transitions	46
2.3.3 Labelled Transition Systems Generated by States	49

vi	Contents	
	2.3.4 Graphical Representation of Labelled Transition Systems	50
	2.3.5 Deriving Multi-step Transitions	52
3	Simple Processes	55
	3.1 Syntax	56
	3.2 Networks	56
	3.2.1 Support and Running Processes	57
	3.2.2 Notation for Networks	57
	3.2.3 Equality of Networks	59
	3.2.4 Properties of Parallel Composition	60
	3.2.5 Representing Networks with Finite Supports	62
	3.3 Semantics	62
	3.3.1 Communication	63
	3.3.2 Parallel Execution	64
	3.4 Fundamental Properties of the Semantics	66
	3.4.1 Transitions and Process Names	66
	3.4.2 Transitions and Process Removal	67
	3.4.3 Transitions and Network Restriction	69
	3.5 Parallelism, Communication Safety, and Starvation-Freedom	71
	3.5.1 Parallelism	71
	3.5.2 Communication Safety	73
	3.5.3 Starvation-Freedom	74
4	Endpoint Projection	76
	4.1 Definition of Endpoint Projection	77
	4.2 Finite Representability of Endpoint Projections	80
	4.3 Correctness of Endpoint Projection	81
	4.3.1 Completeness	83
	4.3.2 Soundness	86
	4.4 Consequences of the Correctness of Endpoint Projection	87
	4.4.1 Communication Safety for Endpoint Projection	87
	4.4.2 Starvation-Freedom for Endpoint Projection	88
Part II	Computation	91
	<i>Introduction to Part II</i>	92
5	Memory and Local Computation	95
	5.1 Stateful Choreographies	95
	5.1.1 Syntax	95
	5.1.2 Semantics	98
	5.2 Stateful Processes	102
	5.2.1 Syntax	102
	5.2.2 Semantics	103
	5.3 Endpoint Projection	104

6	Conditionals and Knowledge of Choice	106
6.1	Conditionals	106
6.1.1	Conditional Choreographies	106
6.1.2	Nondeterminism	111
6.1.3	Conditional Processes	112
6.1.4	Endpoint Projection	115
6.1.5	Limitations of Projectability	119
6.2	Selections	121
6.2.1	Selective Choreographies	121
6.2.2	Selective Processes	124
6.2.3	Endpoint Projection	126
7	Recursion	134
7.1	Choreographies	134
7.1.1	Syntax	134
7.1.2	Expressivity of Recursive Choreographies	135
7.1.3	Well-Formedness	138
7.1.4	Semantics	139
7.2	Processes	147
7.2.1	Syntax	147
7.2.2	Semantics	149
7.3	Endpoint Projection	151
8	Properties of Choreographies and Endpoint Projection	156
8.1	Basic Properties of Process Projection	156
8.2	Properties of Merging and Sequential Composition	157
8.2.1	Fundamental Properties of Merging	157
8.2.2	Ordering Processes by Branchings	157
8.2.3	Sequential Composition	160
8.2.4	Merging and Transitions	160
8.3	Well-Formedness	161
8.3.1	Properties of Well-Formed Choreographies	163
8.4	Correctness of EPP for Recursive Choreographies	164
8.5	Consequences of the Correctness of Endpoint Projection	170
8.5.1	Communication Safety	172
8.5.2	Deadlock-Freedom	173
8.5.3	Starvation-Freedom	174
Part III	Extensions and Variations	177
9	Conservative Extensions	179
9.1	Request-Reply Interactions	179
9.2	Destructuring Messages	181
9.3	Distributed Conditions	182

viii **Contents**

10	Choreographic Choice	184
10.1	From Local to Choreographic Choice	184
10.2	Choreographies	185
10.3	Processes	188
10.4	Endpoint Projection	189
11	Asynchronous Communication	191
11.1	Choreographies	191
11.2	Processes	197
11.3	Endpoint Projection	199
12	Discussion and Further Reading	201
	Solutions to Selected Exercises	213
	<i>References</i>	219
	<i>Index</i>	227

Illustrations

0.1	Choreography compliance	3
1.1	A directed graph representing flight connections between some cities.	16
1.2	An inference system for flights.	17
1.3	An inference system for flights with symmetric connections.	25
1.4	An inference system for weighted flight walks.	26
1.5	An inference system for weighted flight walks, with symmetry but without a connection from New York to Rome.	31
1.6	An alternative inference system for flights.	32
2.1	Syntax of Simple Choreographies.	39
2.2	Semantics of Simple Choreographies.	41
2.3	Graphical representation of the labelled transition system generated by the choreography $p_1 \rightarrow q_1; p_2 \rightarrow q_2; p_3 \rightarrow q_3; \mathbf{0}$.	51
2.4	Inference system for multi-step transitions.	52
2.5	Derivation of the multi-step transition in (2.5).	53
2.6	Admissible rules for multi-step transitions. Rule SINGLE is derivable.	54
3.1	Syntax of Simple Processes.	56
3.2	Semantics of Simple Processes.	63
3.3	Graphical representation of the Its generated by the network in (3.6).	72
4.1	Endpoint projection (EPP).	76
4.2	Labelled transition systems generated by the choreography in Example 4.3 (on the left) and its endpoint projection (on the right). A dashed arrow from a choreography to a network denotes that the latter is the result of applying endpoint projection to the former.	82
4.3	Labelled transition systems generated by the choreography in (4.5) (on the left) and its endpoint projection (on the right).	83
5.1	Syntax of Stateful Choreographies.	96
5.2	Expression evaluation.	99
5.3	Semantics of Stateful Choreographies.	100
5.4	Syntax of Stateful Processes.	102
5.5	Semantics of Stateful Processes.	103
6.1	Syntax of Conditional Choreographies.	107
6.2	Semantics of Conditional Choreographies.	108
6.3	Definition of the \wp operator for choreographies.	109
6.4	Syntax of Conditional Processes.	113
6.5	Semantics of Conditional Processes.	114

x	Illustrations	
6.6	Definition of the \wp operator for processes.	114
6.7	Syntax of Selective Choreographies.	121
6.8	Semantics of Selective Choreographies, rule for selections.	123
6.9	Syntax of Selective Processes.	124
6.10	Semantics of Selective Processes, rule for selections.	125
7.1	Syntax of Recursive Choreographies.	135
7.2	Semantics of Recursive Choreographies.	142
7.3	Syntax of Recursive Processes.	148
7.4	Semantics of Recursive Processes.	150
8.1	Well-formedness for Recursive Choreographies.	162
9.1	A choreography with a distributed condition for a three-party purchase (left side) and its desugaring (right side).	183
10.1	Syntax of Nondeterministic Recursive Choreographies, new term for choices.	185
10.2	Semantics of Nondeterministic Recursive Choreographies, new rules for choices.	186
10.3	Syntax of Nondeterministic Recursive Processes, new term for choices.	188
10.4	Semantics of Nondeterministic Recursive Processes, new rules for choices.	189
10.5	Merging and process projection for Nondeterministic Recursive Choreographies, new rules.	190
11.1	Syntax of Asynchronous Recursive Choreographies, new terms for asynchronous communication.	191
11.2	Semantics of Asynchronous Recursive Choreographies.	193
11.3	Graphical representation of the Its generated by the configuration in Example 11.1.	196
11.4	Semantics of Asynchronous Recursive Processes.	198
A.1	A derivation of $\text{walk}(\text{New York, New York})$ in the system in Figure 1.2.	213

Notations

\hookrightarrow	An exercise for which a solution is provided	9
!	A (possibly) challenging exercise	9
$\text{conn}(A, B)$	There is a direct connection from A to B	16
$\text{walk}(A, B)$	There is a walk from A to B	16
City	The set of cities considered in Chapter 1	17
\triangleq	Definition symbol, read ‘is defined as’	17
\in	Set membership, read ‘is an element of’	17
$\text{size}(\mathcal{D})$	The size of derivation \mathcal{D}	20
$\text{height}(\mathcal{D})$	The height of derivation \mathcal{D}	20
$\mathcal{D}, \mathcal{E}, \mathcal{F}$	Derivations	22
$\mathcal{D} :: p$	The derivation \mathcal{D} , which concludes the proposition p	22
=	Equality, read ‘is equal to’	24
PName	The set of process names	39
p	A process name	39
C	A choreography	39
SimpleChor	The language of Simple Choreographies	39
$\mathbf{0}$	A terminated program, either in a choreographic language or a process language	40, 56
$p \rightarrow q$	The term or label of a communication from p to q	40
\longrightarrow	A transition relation	41
s	A state in a labelled transition system	41
μ	A transition label	41
$s \xrightarrow{\mu} s'$	The state s can make a transition with label μ to the state s'	41
pn	The function that computes the set of process names in a label or term	41, 80
$S_1 \# S_2$	Shortcut to $S_1 \cap S_2 = \emptyset$ (the sets S_1 and S_2 have empty intersection)	43
$S_1 \cap S_2$	The intersection of the sets S_1 and S_2	43
\emptyset	The empty set	43
$s \xrightarrow{\mu}$	The state s can make a transition with label μ	46
$s \not\xrightarrow{\mu}$	The state s cannot make a transition with label μ	46
$\vec{\mu}$	A sequence of transition labels (possibly empty)	46
ϵ	The empty sequence	46

xii **Notations**

$s \xrightarrow{\vec{\mu}} s'$	There is an execution from s to s' with trace $\vec{\mu}$	47
\rightarrow	A multi-step transition relation	47
$s \xrightarrow{\vec{\mu}}$	The state s has the trace $\vec{\mu}$	47
$s \not\xrightarrow{\vec{\mu}}$	The state s does not have the trace $\vec{\mu}$	47
P	A process term	56
$p!$	Send a message to p	56
$p?$	Receive a message from p	56
SimpleProc	The language of Simple Processes	56
N	A network	57
SimpleNet	The set of networks in Simple Processes	57
supp	The function that maps functions to their supports	57
$p[P]$	The atomic network that returns P for p	57
$N \mid M$	The parallel composition of N and M	58
$N \setminus p$	The network obtained by removing p as running process from N	67
\vec{p}	A sequence of process names	69
$N \upharpoonright_{\{p_1, \dots, p_n\}}$	The restriction of N to $\{p_1, \dots, p_n\}$	69
$N \upharpoonright_{p_1, \dots, p_n}$	Shortcut for $N \upharpoonright_{\{p_1, \dots, p_n\}}$	69
$\llbracket C \rrbracket_p$	The projection of choreography C on process p	77
$\llbracket C \rrbracket$	The endpoint projection (EPP) of choreography C	78
Var	The set of (local) variables	95
x	A variable (used in local stores and expressions)	95
Val	The set of (local) values	95
v	A value (used in local stores and expressions)	95
I	An instruction, either in a choreographic or a process language	95
$I; C$	Do I and then C , in a choreographic language	95
$p.e \rightarrow q.x$	Process p communicates the evaluation of e to q , which stores it in variable x	95
e	A (local) expression	95
$p.x := e$	Process p evaluates e and stores the result in x	95
$f(\vec{z})$	The invocation of function f with the evaluations of \vec{z} as arguments	96
f	A function name	96
StatefulChor	The language of Stateful Choreographies	96
σ	A process store	98
PStore	The set of process stores	98
$\sigma[x \mapsto v]$	The store obtained from σ by updating the value for variable x to v	98
Σ	A choreographic store	98
$\Sigma[p.x \mapsto v]$	The choreographic store obtained from Σ by updating the value for variable x of process p to v	98
$\sigma \vdash e \downarrow v$	The expression e is evaluated to v under σ	99
$\langle C, \Sigma \rangle$	A configuration in Stateful Choreographies	100

$\tau @ p$	The label of an internal action at process p	100
$p.v \rightarrow q$	The label of a communication of the value v from p to q	100
$I; P$	Do I and then P , in a process language	102
$p!e$	Send the evaluation of e to p	102
$p?x$	Receive a value from p and store it in x	102
$x := e$	Store the evaluation of e in x	102
StatefulProc	The language of Stateful Processes	102
StatefulNet	The set of networks in Stateful Processes	102
$\langle N, \Sigma \rangle$	A configuration in Stateful Processes	103
if $p.e$ then C_1 else C_2	Run the choreography C_1 if p evaluates e to <i>true</i> , C_2 otherwise	106
<i>true</i>	The Boolean value true	107
<i>false</i>	The Boolean value false	107
ConditionalChor	The language of Conditional Choreographies	107
$\dot{;}$	The sequential composition operator, either for choreographies or for process terms	108
ConditionalProc	The language of Conditional Processes	113
ConditionalNet	The set of networks in Conditional Processes	113
SLabel	The set of all selection labels	121
L	A selection label	121
$p \rightarrow q[L]$	Process p communicates the selection of label L to q	121
SelectiveChor	The language of Selective Choreographies	121
$p \oplus L$	Send to p the selection of label L	124
$p \& \{L_i : P_i\}_{i \in I}$	Receive from p a label L_j , for $j \in I$, and then proceed as P_j	124
SelectiveProc	The language of Selective Processes	124
\sqcup	The merging operator	127
X	A procedure name	134
RecursiveChor	The language of Recursive Choreographies	134
\mathcal{C}	A set of choreographic procedure definitions	134
$X(\vec{p}) = C$	Procedure X has parameters \vec{p} and body C	134
$X(\vec{p})$	Call procedure X with arguments \vec{p}	134, 147
$C[r/s]$	The choreography C , but with r replacing s	139
RecursiveProc	The language of Recursive Processes	147
\mathcal{P}	A set of process procedure definitions	147
$X(\vec{p}) = P$	Procedure X has parameters \vec{p} and body P	147
$P[r/s]$	The process P , but with r replacing s	149
\sqsupseteq	The branching partial order	157
$\langle \mathcal{C}, \{\vec{p}\}, C \rangle \checkmark$	The choreography C is well-formed in the context of \mathcal{C} and involves at most the processes in \vec{p}	161
\forall	Universal quantification, read ‘for each’, ‘for any’, or ‘for all’	163
EnrichedChor	The language of Enriched Choreographies	179
$[C]$	The desugaring, or expansion, of choreography C	179

xiv **Notations**

$p \stackrel{e}{y} \xrightarrow{x} q \cdot \varrho^x \{C\}$	A Request-Reply, where p communicates a request to q and the latter replies after the choreography C is executed	180
if $p_1.e_1$ and \dots and $p_n.e_n$ then C_1 else C_2	A distributed conditional, where C_1 is executed if each condition e_i evaluates to <i>true</i> in the state of its respective process p_i , and C_2 is executed otherwise	182
$C_1 +_p C_2$	Process p chooses between C_1 and C_2	185
$P + Q$	The process that behaves as P or as Q	188
m	A message	191
(p, v)	A message that consists of a value v sent by a process p	192
(p, L)	A message that consists of a label L sent by a process p	192
\vec{m}	A sequence of messages (possibly empty)	192
Queue	The set of message queues	192
K	A messaging state	192
$p.v \rightarrow q!$	The label of an asynchronous transmission of the value v from p to q	192
$p.v \rightarrow q?$	The label of an asynchronous reception by q of the value v from p	192
$p \rightarrow q[L]!$	The label of an asynchronous transmission of the label L from p to q	192
$p \rightarrow q[L]?$	The label of an asynchronous reception by q of the label L from p	192