

Agent-Based Configuration of (Metaheuristic) Algorithms

Dissertation

zur Erlangung des akademischen Grades
Doktor der Naturwissenschaften (doctor rerum naturalium)
im Fach Informatik

ingerichtet an der
Mathematisch-Naturwissenschaftlichen Fakultät II
der Humboldt-Universität zu Berlin

von
Frau Master in Sciences Dagmar Monett Díaz
geboren am 03.04.1969 in Havanna, Kuba

Präsident der Humboldt-Universität zu Berlin
Prof. Dr. Jürgen Mlynek

Dekan der Mathematisch-Naturwissenschaftlichen Fakultät II
Prof. Dr. Uwe Küchler

Gutachter:

1. Prof. Dr. Hans-Dieter Burkhard, Humboldt-Universität zu Berlin
2. Prof. Dr. Egmar Rödel, Humboldt-Universität zu Berlin
3. Prof. Dr. Pedro Larrañaga, University of the Basque Country

ingereicht am: 06.10.2004
Tag der mündlichen Prüfung: 25.02.2005

Berichte aus der Informatik

Dagmar Monett Díaz

**Agent-Based Configuration of
(Metaheuristic) Algorithms**

Shaker Verlag
Aachen 2005

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the internet at <http://dnb.ddb.de>.

Zugl.: Berlin, Humboldt-Univ., Diss., 2005

Copyright Shaker Verlag 2005

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without the prior permission of the publishers.

Printed in Germany.

ISBN 3-8322-4010-1

ISSN 0945-0807

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen

Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9

Internet: www.shaker.de • eMail: info@shaker.de

Zusammenfassung

Diese Arbeit stellt einen neuen, flexiblen Ansatz für die Konfiguration von Algorithmen vor, der besonders die agentenbasierte Konfiguration metaheuristischer Algorithmen berücksichtigt. Metaheuristiken sind heuristische Suchverfahren, die gute Lösungen durch das gezielte Ändern einer Menge von aktuellen Lösungen finden sollen. Sie sind in vielen Anwendungsgebieten erfolgreich eingesetzt worden. Metaheuristiken sind Beispiele für Algorithmen, in denen Parameter so gut wie möglich angepasst werden müssen, wie z.B. genetische Parameter in der Domäne der evolutionären Algorithmen. Allerdings ist die Auswahl der anzupassenden Parameter sowie die anschließende Anpassung und Spezifikation dieser Parameter für konkrete Problemstellungen oft kompliziert. Beides sind anerkannt schwierige Aufgaben im Forschungsgebiet der Metaheuristiken. Idealerweise sind Metaheuristiken selbst anpassungsfähig. Gewöhnlich wird aber die Anpassung von Steuerparametern, die von Natur aus zeitaufwendig ist, durch unzulängliche experimentelle Methoden oder *brute-force*-Mechanismen ausgeführt. Einige wenige Methoden zur Konfiguration metaheuristischer Algorithmen sind in der Literatur schon vorgeschlagen worden. Diese sind aber noch nicht für ein verteiltes Problemlösen geeignet. Andererseits gibt es zu dem Problem der Anpassung von Metaheuristiken sehr wenig publizierte Forschungsergebnisse. Alles in allem bleibt dieses Problem also eine aktuelle offene Forschungsfrage.

Die vorliegende Arbeit beschreibt einen agentenbasierten Ansatz zur Konfiguration von Algorithmen. Der Hauptfokus des Ansatzes, die +CARPS-Architektur (+CARPS: *Multi-Agent System for Configuring Algorithms in Real Problem Solving*), liegt darin, agentenbasierte Verfahren für die Anpassung algorithmischer Parameter anzuwenden. Vor allem wird die Parameterkonfiguration (d.h. die Anpassung der unbekannt Parameter) mittels verteilter, autonomer, kooperierender und spezialisierter Agenten erreicht. Die Hauptidee hinter dem vorgeschlagenen Ansatz liegt in der Vorstellung, Agenten mit der Fähigkeit zur "Anpassung metaheuristischer Parameter" in der Art auszustatten, wie dies ein Experte tun würde. Die agentenbasierte Konfiguration von Algorithmen ist ein passender, robuster und wirksamer Ansatz, der selbstständig das Verfahren der Konfiguration durchführt, wobei die Agenten die Algorithmen automatisch ausführen und konfigurieren.

Die Agenten führen ein Optimierungsverfahren aus. Es handelt sich hier-

bei um ein Verfahren, das Parameterkonfigurationen optimiert. Anfänglich werden Ausgangskonfigurationen, die Startwerte für die freien Parameter enthalten, erzeugt. Dann werden weitere notwendige Initialisierungen ausgeführt, z.B. jene der Agentenkommunikation. Anschließend werden neue Konfigurationen, die näher optimiert werden sollen, mit einem dazu angewendeten Konfigurationsalgorithmus erzeugt. Innerhalb dieses auf kooperativ interagierenden Agenten basierenden Verfahrens, werden potentiell gute Lösungen zwischen den Agenten ausgetauscht. Dies ermöglicht den Agenten jeweils eine neue Perspektive auf den Suchraum. Dieser Austausch von Zwischenergebnissen ist ein geeigneter und wesentlicher Schritt, lokale Optima zu vermeiden. Schließlich werden die besten Lösungen, die gefunden wurden, als Lösung des Konfigurationsprozesses präsentiert.

Die Dissertationsschrift umfaßt drei Hauptteile. Jeder Teil besteht aus mehreren Kapiteln. Der erste Teil beschäftigt sich mit den aus der Literatur bekannten metaheuristischen Ansätzen, sowie mit Gebieten der Künstlichen Intelligenz, die diese Arbeit unterstützen. Die Klassifikation der Beziehungen zwischen Metaheuristiken und Künstlicher Intelligenz ist ebenfalls eingeschlossen, wobei verschiedene Mechanismen beider Bereiche und deren Interaktion vorgestellt werden. Im zweiten Teil, dem Schwerpunkt der Arbeit, wird sowohl die Theorie über die Konfiguration von Metaheuristiken eingeführt, als auch ihre agentenbasierte Automatisierung beschrieben. Die Konzeption, Struktur, Implementierung und Realisation des Multiagenten-Systems, die +CARPS-Architektur, werden innerhalb dieses Teiles der Dissertation behandelt. Im dritten und letzten Teil werden verschiedene Experimente vorgestellt und die praktische Umsetzung des agentenbasierten Ansatzes zusammengefaßt.

+CARPS-Agenten entsprechen die FIPA-Spezifikationen und sind kooperierende JADE-Agenten, die in Java entwickelt wurden. Die stärksten Gründe für die Auswahl der JADE-Plattform waren ihre durchgängige Entwicklungsunterstützung und die anerkannten Vorteile als ein Projekt mit offenem Quell-Code, das den neuesten FIPA-Spezifikationen folgt. Die verteilte Agentenplattform, der Nachrichten-Transportmechanismus und die graphischen Werkzeuge zum Test der Multiagenten-Systeme, um ein paar Eigenschaften zu nennen, waren ebenfalls mitentscheidend.

+CARPS besteht aus verschiedenen Arten von Agenten, die die autonome Konfiguration von metaheuristischen Algorithmen unterstützen. Unter den wichtigsten Tätigkeiten der Agenten sind die folgenden: *user mediators* leiten den Benutzer bei der Anwendung von +CARPS an und helfen ihm bei der Auswahl und Vorgabe von Daten. Sie präsentieren die Ergebnisse in einer dem Anwender verständlichen Weise; *starting configuration builders* und *instantiation strategy managers* bestimmen die Werte für die notwendigen Parameter entweder bei den am Start erzeugten Konfigurationen oder während deren Anpassung; *algorithm configurators* überwachen und steuern den Ablauf des Konfigurationsalgorithmus und wählen passende Parameter für die nächsten Kalkulationen; *algorithm solvers*

führen selbständig die Algorithmen aus, rufen Daten ab und bearbeiten die erhaltenen Ergebnisse; und *solution managers* kontrollieren die Organisation und Klassifizierung der erhaltenen Lösungen, um sie den Benutzern zu präsentieren. Alle Agenten, die in der Architektur mit eingeschlossen werden, sowie ihre Kennzeichnung, Kommunikation und Interaktionsprotokolle werden beschrieben. Die Agenten erlauben es dem Benutzer, verschiedene Funktionen zu delegieren, die er andernfalls allein durchführen müßte. Der Anwender wird dabei weder mit der Komplexität der Algorithmen konfrontiert noch mit der Konfiguration und Anpassung der notwendigen Parameter.

Die Hauptbeiträge dieser Dissertation sind im Folgenden aufgelistet:

- Der Entwurf, die Entwicklung und die Implementierung eines Software-Prototyps, der die Konfiguration von (metaheuristischen) Algorithmen auf der Basis autonomer Agenten unterstützt. +CARPS, der agentenbasierte Ansatz zur Konfiguration von Algorithmen, einschließlich der zu Metaheuristiken, ist beschrieben, implementiert und getestet.
- Die theoretische Beschreibung und die Formalisierung des Konfigurationsproblems, welche die verbesserte Spezifikation und den Entwurf der verschiedenen Agenten sowie deren grundlegender Arbeitsprinzipien und Verhaltensweisen ermöglicht haben.
- Die Implementierung eines *Random-Restart Hill-Climbing* Algorithmus, den einige spezialisierte Agenten während des Konfigurationsverfahrens verwenden, um die Suche nach optimalen Konfigurationen zu automatisieren.
- Die Entwicklung und die Implementierung von neuen Interaktionsprotokollen, die den Kommunikationsbedingungen der Agenten in der Analyse-domäne entsprechen.
- Die Konzeption und die Implementierung von Ontologien, Interaktionsprotokollen, Agentenverhalten, und verschiedenen Java-Klassen zur Unterstützung der Infrastruktur, die die agentenbasierte Konfiguration von Algorithmen benötigt.

+CARPS ist auch ein Mehrzwecksystem, in dem Steuerfaktoren von Algorithmen leicht überwacht werden können, was, zum Beispiel für Forscher, die das Verhalten der Parameter in Metaheuristiken studieren wollen, nützlich sein kann. Gleichzeitig ist +CARPS ein mächtiges und nützliches Werkzeug beim Durchführen von Experimenten mit diesen Algorithmen, sowie ein geeigneter Rahmen für das Monitoring der Experimenten.

Die Funktionalität der in dieser Arbeit vorgestellten Agentenarchitektur wird anhand einer Vielzahl von Experimenten veranschaulicht. Als Beispiele praktischer Anwendungen gelang bereits die Schätzung kinetischer Parameter in Prozessen der Copolymerisation mit Hilfe genetischer Algorithmen. Copolymerisationen bieten die Möglichkeit, Eigenschaften von Biomaterialien zu beeinflussen,

mit deren Hilfe die Entwicklung neuer biomedizinischer Substanzen unterstützt werden soll. Außerdem hat die Anpassung einer Evolutionsstrategie dazu geführt, wichtige Aspekte des agentenbasierten Konfigurationsprozesses optimieren zu können.

Der wichtiger Beitrag dieser Dissertation ist die Umsetzung eines agentenbasierten Ansatzes zur Algorithmenkonfiguration durch die +CARPS-Architektur, der eine neue und sinnvolle Alternative zu anderen Ansätzen bildet und der auf verteilten, kollaborativen JADE-Agenten beruht. Die Ergebnisse dieser Dissertation sind nicht nur wegen der vorgestellten konkreten +CARPS Architektur von Bedeutung, sondern auch wegen der konzipierten und implementierten Bausteine für eine verteilte, automatisierte und autonome Konfiguration von Algorithmen.

Abstract

An agent-based approach to the configuration of algorithms including, but not limited to, metaheuristics is proposed in this work. Metaheuristics are algorithmic techniques that look for good solutions by changing a current set of solutions from the search space. They are examples of algorithms where parameters need to be set up as good as possible, like genetic parameters in the Evolutionary Computation domain. Usually, control parameters are set by hand or in the spirit of brute-force mechanisms, which are inherently time-consuming. Yet the selection of the most adequate parameter values is a recognized arduous work in the metaheuristics community; furthermore, not all metaheuristics are auto-adaptive. Some methods for configuring metaheuristic algorithms have been already proposed in the literature. However, the problem of configuring metaheuristics continues being a very difficult problem with very few published research works, and therefore it remains a current open question.

Instead of requiring that users fine-tune metaheuristics' parameters by experimenting, which is costly in human and time resources, this thesis proposes for that purpose an agent-based approach, which is guided by cooperation among agents in a multiagent system. The agents carry out an optimization process: the process of configuring the metaheuristics through the fine-tuning of their parameters. The application of agent-based methods from Artificial Intelligence emerges as a valuable approach because it allows users and developers to delegate functions and efforts to the computers, thus automating or *semi*-automating the configuration of metaheuristics without a human supervision. Consequently, this thesis aims also to develop a flexible and easy-to-use multiagent system for these purposes, the practical aspects being its major contributions.

In pursuing these aims, a multiagent system (i.e. +CARPS: *Multi-Agent System for Configuring Algorithms in Real Problem Solving*) based on collaborative, distributed, FIPA-compliant JADE agents was designed, implemented and tested. +CARPS consists of different types of cooperatives agents that concurrently support the autonomous configuration of metaheuristic algorithms. Among the most important aspects concerning +CARPS agents' activities are the following: *user mediators* interact with the users to retrieve from the system data related to the problems to be solved, to the metaheuristics to be configured, among other information; *starting configuration builders* and *instantiation*

strategy managers determine the values for the parameters to fine-tune by constructing initial configurations and by applying different instantiation strategies, respectively; *algorithm configurators* apply optimization procedures to obtain and improve solutions to the configuration problem; *algorithm solvers* run the metaheuristics with the appropriate input information and retrieve results when their execution is finished; and *solution managers* control the process of organizing and classifying obtained solutions. All types of agents included in the architecture are introduced, as well as their characteristics, communication, and interaction protocols.

Other important contributions of this work are to provide the needed infrastructure to support the agent-based configuration of metaheuristics, as well as to develop a framework in which monitoring of their control factors becomes an easy task. At the same time, the agent architecture can be seen as a powerful tool, useful for conducting experiments when executing metaheuristic algorithms, but, in the first place, as a suitable approach that makes use of agent technology issues to contribute to the configuration of algorithms.

Acknowledgements

*“Wir können nicht genug anerkennen,
wie nötig Mitteilung, Beihilfe, Erinnerung und Widerspruch sei,
um uns auf dem rechten Weg zu erhalten
und vorwärts zu bringen.”*
– GOETHE

This doctoral thesis is the result of years of work and studies. Its materialization would not be possible without the support of many people, whose advice and encouragement allowed me to start it and finish it. I want to express my gratitude to all of them here.

My first very special thanks and love go to my parents, my brother, who revised different releases of this thesis, and my family in Cuba, for whom the long separation affected and affects them and me, but never stopped them from encouraging me to walk along this difficult road. My deepest love and thanks also go to my husband, Abel Martín, who patiently trusted me and unconditionally accompanied me during this journey.

A deep gratitude goes also to my supervisor and leader of the Artificial Intelligence Group at Computer Science Department, Humboldt University of Berlin, Prof. Hans-Dieter Burkhard, who gave me the possibility to come to Berlin to do my research and supervised it. I will be always grateful for his advice and helpful contributions during my research.

I appreciate very much the support of Dr. Lylia Esther Cuesta and Dr. Mathias Geldbrich for their exceptional help while revising different versions of the thesis, for their advice, for their friendship.

I want to thank all colleagues and PhD students from the Computer Science Department at the Humboldt University of Berlin with whom I directly or indirectly worked, as well as to those who helped me over my stay. Special mention to Dr. Gabriela Lindemann, who gave me her continuous support since my first visit to Berlin in 1997 and who revised part of the thesis, and to Mirjam Minor also for her comments and for her support.

I also want to thank Prof. Ingo Rechenberg and other colleagues from the Technical University of Berlin for allowing me to present my research at their group and for the helpful discussions that followed.

Thanks also goes to the main sponsors of this research, the German Academic Exchange Service (DAAD), the German Foundation Rosa Luxemburg, and the Woman Support Program from the Humboldt University of Berlin for their scholarships and financial support.

Last but not least, I want to thank all those who never stopped cheering me up and giving me their emotional support, as well as those who helped me during my stay in Germany.

To all of you, thank you very much for allowing me to make this dream come true.

Contents

List of Figures	viii
List of Tables	ix
1 Introduction	1
1.1 Configuration of Metaheuristics	1
1.1.1 Methods for configuring metaheuristics	2
1.2 Why agents? Preliminary comments	3
1.3 Scope and Goals of the Thesis	5
1.4 Contributions of the Thesis	8
1.5 Outline of the Thesis	8
I Background	11
2 Metaheuristics at a Glance	13
2.1 Introduction	13
2.2 Overview and general aspects	14
2.3 Classification of Metaheuristics	15
2.4 Summary	18
3 Artificial Intelligence at a Glance	19
3.1 Introduction	19
3.2 Distributed Artificial Intelligence	20
3.2.1 Agents	21
3.2.2 Multiagent systems	23
3.2.3 Distributed Problem Solving	24
3.3 Summary	26
4 Related Work	27
4.1 Introduction	27
4.2 Setting up parameters in metaheuristics	27
4.3 Approaches that configure metaheuristics	30
4.3.1 Meta-evolution as configurator	30

4.3.2	Parallel approaches	32
4.3.3	More general configurators	32
4.4	Testing and developing metaheuristics	37
4.5	Metaheuristics and agent-based techniques	38
4.5.1	Related examples	40
4.6	Summary	42

II Multiagent System for Configuring Algorithms in Real Problem Solving 43

5	Formal Concepts Related to Metaheuristics and their Configuration	45
5.1	Introduction	45
5.2	Parameters and configurations	45
5.2.1	Design of experiments and evaluation	47
5.2.2	Performance indexes	49
5.3	Worth of a metaheuristic	50
5.3.1	Stop criteria	50
5.3.2	Definition of the worth	53
5.4	The metaheuristic configuration problem	56
5.4.1	Pareto-optimal configurations	56
5.4.2	Configuration problem	58
5.5	Summary	59
6	Agent-based Configuration of Metaheuristics. +CARPS's Design	61
6.1	Introduction	61
6.1.1	Decomposition of the configuration problem	61
6.1.2	<i>Why agents?</i> revisited	62
6.2	Agent architecture: +CARPS	64
6.2.1	User Mediator (UM agent)	66
6.2.2	Starting Configuration Builders (SCB agents)	67
6.2.3	Instantiation Strategy Manager (ISM agent)	71
6.2.4	Algorithm Solvers (AS agents)	74
6.2.5	Algorithm Configurators (AC agents)	75
6.2.6	Solution Manager (SM agent)	81
6.3	Interaction protocols in +CARPS	82
6.3.1	The Helper-Booker Interaction Protocol	82
6.3.2	The Engagement Interaction Protocol	84
6.3.3	The AC-AS Request Interaction Protocol	85
6.4	Basic essentials of agents creation and communication in +CARPS	87
6.5	Summary	88

7	Implementation of +CARPS	89
7.1	Introduction	89
7.2	Making +CARPS agents talk and reason	90
7.2.1	+CARPS vocabularies and ontologies	91
7.3	+CARPS agents' communications	97
7.3.1	User Mediator (UM agent)	97
7.3.2	Starting Configuration Builders (SCB agents)	106
7.3.3	Instantiation Strategy Manager (ISM agent)	108
7.3.4	Algorithm Solvers (AS agents)	111
7.3.5	Algorithm Configurators (AC agents)	115
7.3.6	Solution Manager (SM agent)	122
7.4	Interaction Protocols	123
7.4.1	The Helper-Booker Interaction Protocol	123
7.4.2	The Engagement Interaction Protocol	128
7.4.3	The AC-AS Request Interaction Protocol	132
7.5	Some design and technical information	134
7.6	Summary	135
8	Case Studies	137
8.1	Introduction	137
8.2	Basic essentials of evolutionary algorithms	138
8.2.1	Genetic Algorithms	139
8.2.2	Evolution Strategies	141
8.3	GA in Chemical Kinetics	141
8.3.1	Inverse problems	141
8.3.2	Applying GA	146
8.4	Summary	154
III Results, Discussion, and Conclusions of the Thesis		155
9	Testing +CARPS. Discussion	157
9.1	Introduction	157
9.2	Experimental part	159
9.2.1	General assumptions	161
9.2.2	Test problems for the GA	162
9.2.3	Test problem for the ES	163
9.3	Results	163
9.3.1	Presenting general aspects from the +CARPS fine-tuning	164
9.3.2	Varying the equations to calculate the worth	170
9.3.3	Varying the weight vectors	174
9.3.4	Varying the number of neighbors	176
9.3.5	Considering experimental repetitions	181

9.3.6	Varying the proportion of AC agents	182
9.3.7	Considering more search trials	187
9.4	Summary	193
10	Conclusions and Outlook	195
10.1	Conclusions and summary of the main contributions	195
10.2	Open questions	197
10.2.1	Comments on possible extensions	197
10.3	+CARPS applications	199
10.4	Final conclusions	200
A	JADE	201
A.1	What is JADE?	201
A.2	Where to find it?	201
A.3	General aspects about the platform	201
A.4	Related materials and tutorials	201
B	Format of the Input Files to the Metaheuristics from the Test Cases	203
B.1	Introduction	203
B.2	Format of the input files for the GA	203
B.2.1	Format of the input data file containing metaheuristic's parameters	204
B.2.2	Format of the input data file containing real problem information	205
B.3	Format of the input file for the ES	206
B.3.1	Format of the input data file containing metaheuristic's parameters	206
	Bibliography	207
	Index	229

List of Figures

1.1	Optimization and simulation processes in the agent-based configuration of metaheuristics.	4
1.2	System architecture and user interfaces: general view.	5
4.1	Tree Growing and Pruning Method for configuring metaheuristic algorithms.	33
4.2	Factorial Experimental Design and Local Search-based Method for configuring heuristic algorithms.	34
4.3	F-Race algorithm for configuring metaheuristics.	35
4.4	Fractional Experimental Design and Local Search-based Method for configuring metaheuristic algorithms.	36
4.5	Relationships between both metaheuristics and agent-based techniques.	40
5.1	Example of fixed and free parameters in the configuration of a metaheuristic.	47
6.1	Agent-Based Configuration of (Metaheuristic) Algorithms.	63
6.2	Layered view of +CARPS.	64
6.3	+CARPS architecture: Main interactions among the agents.	65
6.4	UM agent and its interactions.	67
6.5	SCB agents and their interactions.	68
6.6	Example of SCB agent. Four different starting configurations are constructed, \mathbf{p}_1 , \mathbf{p}_2 , \mathbf{p}_3 , and \mathbf{p}_4 , depending on the instantiation strategies that are available.	70
6.7	ISM agent and its interactions.	71
6.8	AS agents and their interactions.	74
6.9	Input and output data files.	75
6.10	AC agents and their interactions.	75
6.11	Agent-Based Random-Restart Hill-Climbing: Search procedure to improve configurations.	78
6.12	SM agent and its interactions.	81
6.13	Sequence diagram for the Helper-Booker Interaction Protocol.	83

6.14	Sequence diagram for the Engagement Interaction Protocol.	85
6.15	FIPA Request Interaction Protocol.	86
7.1	Classes diagram for both the +CARPS <i>Configuration</i> vocabulary and its ontology.	92
7.2	Classes diagram for both the +CARPS <i>User Problem</i> vocabulary and its ontology.	95
7.3	Classes diagram for both the +CARPS <i>Instantiation Strategy</i> vocabulary and its ontology.	95
7.4	Classes diagram for both the +CARPS <i>Agent's List</i> vocabulary and its ontology.	96
7.5	Behaviors of the UM agent.	97
7.6	Graphical User Interface of +CARPS.	99
7.7	Parameter Frame from +CARPS.	99
7.8	Agents and Configuration Algorithms Frame from +CARPS.	100
7.9	Tuning Frame from +CARPS.	101
7.10	User Problem Frame from +CARPS.	101
7.11	Behaviors of the SCB agents.	107
7.12	Behaviors of the ISM agent.	108
7.13	Behaviors of the AS agents.	112
7.14	Finite State Machine from AS agents for running algorithms.	114
7.15	Behaviors of the AC agents.	116
7.16	Finite State Machine for the ABRRH algorithm.	118
7.17	Behaviors of the SM agent.	122
7.18	Helper-Booker Initiator Finite State Machine.	124
7.19	Helper-Booker Responder Finite State Machine.	126
7.20	Engagement Initiator Finite State Machine.	129
7.21	Engagement Responder Finite State Machine.	130
7.22	Request Initiator Finite State Machine.	132
8.1	Pseudo-code of a Genetic Algorithm.	140
8.2	GA classes hierarchy.	147
8.3	Functioning of the starting method.	148
8.4	Main GA cycle.	150
8.5	Random generation of individuals in a population.	152
8.6	Recombination in a population of individuals.	153
8.7	Functioning of the simple one-point crossover operator.	153
9.1	Examples of agents, containers, and platforms.	159
9.2	Sniffer agent from JADE.	160
9.3	Singular and Pareto-optimal meta-solutions found by AC1 (left) and AC2 (right) agents for the system Q9-VP according to different criteria.	166

9.4	Partial and Pareto-optimal meta-solutions found by AC1 (left) and AC2 (right) agents for the system Q9-VP according to different criteria.	167
9.5	All singular meta-solutions found by both AC1 and AC2 agents for the system Q9-VP according to different criteria.	168
9.6	Worth of partial meta-solutions found by AC1 (left) and AC2 (right) agents for the system Q9-VP.	168
9.7	Worths of partial meta-solutions found by both AC1 and AC2 agents for the system Q9-VP.	169
9.8	User problem-related results for the system Q9-VP.	170
9.9	Variation of the solution quality when using different worth equations.	171
9.10	Quality of both the most significant singular meta-solutions and the Pareto-optimal ones with respect to the number of function evaluations.	172
9.11	Elapsed time of interaction protocols for the experiment <i>E-6&8a</i>	173
9.12	Pareto sets that are obtained when varying the weight vector.	175
9.13	Variation of the solution quality when using different weight vectors.	176
9.14	Elapsed time of interaction protocols for the experiment <i>E-8a&d</i>	176
9.15	Sets of singular meta-solutions obtained with respect to the time (left) and to the number of function evaluations (right) for two and eight neighbors.	178
9.16	Configurations that are generated when varying the number of neighbors.	179
9.17	Configurations that are generated for four (left) and eight (right) neighbors, near the optimum values.	180
9.18	Variation of the solution quality when changing the number of neighbors.	180
9.19	Elapsed time of both the ABRRH configuration algorithm and the user problem solver when varying the number of neighbors.	181
9.20	Elapsed time in milliseconds of both the ABRRH configuration algorithm and the user problem solver for two neighbors after five repetitions.	182
9.21	Configurations that are generated (left) and sets of singular meta-solutions (right) when varying the proportion of AC agents per parameter to fine-tune.	184
9.22	Variation of the solution quality (left) and the worth (right) when changing the proportion of AC agents per parameter to fine-tune.	185
9.23	Elapsed time of the Engagement IP when varying the proportion of AC agents per parameter to fine-tune.	186
9.24	Elapsed time in milliseconds of both the ABRRH configuration algorithm and the user problem solver, when varying the proportion of AC agents per parameter to fine-tune.	187

9.25	Distribution of singular meta-solutions over the analysis intervals.	188
9.26	Solution qualities for varying control parameters.	189
9.27	Solution qualities in the whole interval (left) and zoomed in (right) for both control parameters.	190
9.28	Distribution of Pareto-optimal singular meta-solutions over the analysis intervals.	190
9.29	Pareto-optimal singular meta-solutions according to their qualities.	191
9.30	Pareto-optimal singular meta-solutions and the parameter values that they represent.	191
9.31	Elapsed time of the Engagement IPs.	192
9.32	Elapsed time of both the ABRHHC configuration algorithm and the user problem solver.	192

List of Tables

7.1	Lists of agents	104
7.2	States from the ABRRH algorithm and their abbreviations	119
7.3	+CARPS technical information	134
8.1	Some configuration methods and the algorithms that they configure.	137
8.2	Some disadvantages when using optimization methods.	145
8.3	GA technical information	154
9.1	Observed data for the system Q9-VP.	162
9.2	Settings for the experiment <i>E-q9vp</i>	164
9.3	GA fixed parameters for the experiment <i>E-q9vp</i>	164
9.4	Settings for the experiment <i>E-6&8a</i>	171
9.5	Pareto-optimal meta-solutions from the experiment <i>E-6&8a</i>	173
9.6	Settings for the experiment <i>E-8a&d</i>	175
9.7	Settings for the experiment <i>E-1prop&n</i>	177
9.8	Settings for the experiment <i>E-4n&ACprop</i>	183
9.9	Settings for the experiment <i>E-60t</i>	188
9.10	Statistics for singular meta-solutions, experiment <i>E-60t</i>	188
9.11	Pareto-optimal singular meta-solutions from the experiment <i>E-60t</i>	191