

A Dissertation submitted in fulfilment of the
requirements for the degree of Doctor of
Philosophy

New Information Model that Allows Logical Distribution of the Control Plane for Software-Defined Networking

**The Distributed Active Information Model (DAIM) can enable
an effective distributed control plane for SDN with OpenFlow as
the standard protocol**

Pakawat Pupatwibul

Autumn 2016

University of Technology Sydney
Faculty of Engineering and Information Technology
Centre for Real Time Information Networks

Supervisor

Professor Robin Braun

Co-supervisor

Dr. Bruce Moulton

Date of the graduation

May 2016

I dedicate this thesis to my lovely father, mother, sister,
beloved wife and sons for their love and support.

CERTIFICATE OF ORIGINAL AUTHORSHIP

I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text.

I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Signature of Student:

Date:

Abstract

In recent years, technological innovations in communication networks, computing applications and information modelling have been increasing significantly in complexity and functionality driven by the needs of the modern world. As large-scale networks are becoming more complex and difficult to manage, traditional network management paradigms struggle to cope with traffic bottlenecks of the traditional switch and routing based networking deployments. Recently, there has been a growing movement led by both industry and academia aiming to develop mechanisms to reach a management paradigm that separates the control plane from the data plane.

A new emerging network management paradigm called Software-Defined Networking (SDN) is an attempt to overcome the bottlenecks of traditional data networks. SDN offers a great potential to ease network management, and the OpenFlow protocol in particular is often referred to as a radical new idea in networking. SDN adopts the concept of programmable networks which separate the control decisions from forwarding hardware and thus enabling the creation of a standardised programming interface. Flow computation is managed by a centralised controller with the switches only performing simple forwarding functions. This allows researchers to implement their protocols and algorithms to control data packets without impacting on the production network. Therefore, the emerging OpenFlow technology provides more flexible control of networks infrastructure, are cost effective, open and programmable components of network architecture.

SDN is very efficient at moving the computational load away from the forwarding plane and into a centralised controller, but a physically centralised controller can represent a single point of failure for the entire network. This centralisation approach brings optimality, however, it creates additional problems of its own including single-domain restriction, scalability, robustness and the ability for switches to adapt well to changes in local environments.

This research aims at developing a new distributed active information model (DAIM) to allow programmability of network elements and local decision-making processes that will essentially contribute to complex distributed networks. DAIM offers adaptation algorithms embedded with intelligent information objects to be applied to such complex systems. By applying the DAIM model and these adaptation algorithms, managing complex systems in any distributed network environment can become scalable, adaptable and robust. The DAIM model is integrated into the SDN architecture at the level of switches to provide a logically distributed control plane that can manage the flow setups. The proposal moves the computational load to the

switches, which allows them to adapt dynamically according to real-time demands and needs. The DAIM model can enhance information objects and network devices to make their local decisions through its active performance, and thus significantly reduce the workload of a centralised SDN/OpenFlow controller.

In addition to the introduction (Chapter 1) and the comprehensive literature reviews (Chapter 2), the first part of this dissertation (Chapter 3) presents the theoretical foundation for the rest of the dissertation. This foundation is comprised of the logically distributed control plane for SDN networks, an efficient DAIM model framework inspired by the O:MIB and *hybrid O:XML* semantics, as well as the necessary architecture to aggregate the distribution of network information. The details of the DAIM model including design, structure and packet forwarding process are also described.

The DAIM software specification and its implementation are demonstrated in the second part of the thesis (Chapter 4). The DAIM model is developed in the C++ programming language using free and open source NetBeans IDE. In more detail, the three core modules that construct the DAIM ecosystem are discussed with some sample code reviews and flowchart diagrams of the implemented algorithms. To show DAIM's feasibility, a small-size OpenFlow lab based on Raspberry Pi's has been set up physically to check the compliance of the system with its purpose and functions. Various tasks and scenarios are demonstrated to verify the functionalities of DAIM such as executing a ping command, streaming media and transferring files between hosts. These scenarios are created based on OpenVswitch in a virtualised network using *Mininet*.

The third part (Chapter 5) presents the performance evaluation of the DAIM model, which is defined by four characteristics: round-trip-time, throughput, latency and bandwidth. The ping command is used to measure the mean RTT between two IP hosts. The flow setup throughput and latency of the DAIM controller are measured by using *Cbench*. Also, *Iperf* is the tool used to measure the available bandwidth of the network. The performance of the distributed DAIM model has been tested and good results are reported when compared with current OpenFlow controllers including NOX, POX and NOX-MT. The comparisons reveal that DAIM can outperform both NOX and POX controllers. The DAIM's performance in a physical OpenFlow test lab and other parameters that can affect the performance evaluation are also discussed.

Because decentralisation is an essential element of autonomic systems, building a distributed computing environment by DAIM can consequently enable the development of autonomic management strategies. The experiment results show the DAIM model can be one of the architectural approaches to creating the autonomic service management for SDN. The DAIM model can be utilised to investigate the functionalities required by the autonomic networking within the ACNs community. This efficient DAIM model can be further applied to enable adaptability and autonomy to other distributed networks such as WSNs, P2P and Ad-Hoc sensor networks.

Contents

Abstract	i
Nomenclature	xiii
Acknowledgments	xvii
Related Publications	xix
I. Elaborating on the “Propositions”	1
1. Introduction	3
1.1. Introduction	3
1.2. Background of Network Management Complexity	3
1.3. Research Motivations	5
1.3.1. Motivation from Management of Distributed Complex Networks	5
1.3.2. Motivation from Self-Management Strategies	5
1.3.3. Motivation from OpenFlow-Based SDN	6
1.4. Research Objectives and Scope	6
1.4.1. Research Objectives	7
1.4.2. Research Scope	7
1.5. Problem Statement	9
1.5.1. Research Questions	10
1.5.2. Propositions Derived from the Research Questions	10
1.6. Approach and Methodology	11
1.6.1. Reviewing the Scholarly Literature	11
1.6.2. Designing a Candidate System	12
1.6.3. Emulating the Candidate System Using Mininet	13
1.6.4. Implementing the Candidate System	15
1.6.5. Validating the Candidate System	16
1.6.6. Concluding Observations	16
1.7. Outline of the Thesis	17
1.8. Statement of Contributions	19
2. Background and Literature Review	23
2.1. Introduction	23

2.2.	Network Management Background	24
2.2.1.	Five ISO Functional Areas in Network Management	25
2.2.2.	Network Management Protocols	27
2.2.3.	Centralised Management Paradigm	29
2.2.4.	The Needs of Distributed Systems	31
2.3.	Standard Sets of Information Model	34
2.3.1.	Common Information Model (CIM)	35
2.3.2.	Shared Information and Data model (SID)	35
2.3.3.	Limitations of CIM and SID	36
2.3.4.	Current Information Models vs. Proposed DAIM Model	37
2.4.	Software Defined Networking (SDN)	37
2.4.1.	Overview of OpenFlow-Based SDN	41
2.4.2.	Packet Processing in OpenFlow	43
2.4.3.	OpenFlow Switch	44
2.4.4.	OpenFlow Controller	47
2.4.5.	OpenFlow Channel and Protocol	53
2.4.6.	SDN Development Tools	56
2.5.	SDN Scalability Issues	59
2.6.	Related Work to Solve OpenFlow Scalability Issues	61
2.6.1.	Optimisation Techniques	62
2.6.2.	Devolving Some Control Functions Back to the Switches	62
2.6.3.	Designing a Distributed Control Platform	64
2.7.	Autonomic Communications	66
2.7.1.	Background of Autonomic Communications	66
2.7.2.	Overview of Self-X Properties	69
3.	Distributed Active Information Model Theory	73
3.1.	Introduction	73
3.2.	Theoretical Framework	74
3.2.1.	O:MIB Theory	74
3.2.2.	Use of O:XML	78
3.2.3.	Using DAIM as a Logically Distributed Control Plane	80
3.3.	DAIM Model Paradigm	83
3.3.1.	Objectives of Designing DAIM	84
3.3.2.	DAIM Model Architecture	85
3.3.3.	DAIM Agents Implementation	86
3.3.4.	Uniqueness of DAIM Model	87
3.4.	Packet Processing Within DAIM	88
3.5.	Risk Scenarios of the DAIM Model	91

II. Proving the “Propositions” **93**

4. Integrating DAIM to OpenFlow-Based SDN Using Mininet Emulator **95**

4.1. Introduction	95
4.2. DAIM Model Implementation	96
4.2.1. Phase 1: Basic Carrier Functionality	97
4.2.2. Phase 2: Semi-Distributed Functionality	98
4.2.3. Phase 3: Fully Distributed Functionality	99
4.3. DAIM Software Specification	101
4.3.1. Overview of Model	101
4.3.2. The Communication Module	104
4.3.3. The Local Storage Module	109
4.3.4. The Controller Module	113
4.4. Setup Requirements for Testing DAIM	119
4.4.1. Scenarios for Testing DAIM	120
4.5. DAIM System Validation	123
4.5.1. Communication Example	123
4.5.2. Flow Table Buildup with Example of Ping Traffic	125
4.5.3. Creating a Linux Command Line Chat Server	128
4.5.4. Network Streaming via VLC Media Player	130
4.5.5. Run a Simple Web Server and Client	131

5. DAIM Performance Results and Evaluation **133**

5.1. Introduction	133
5.2. Test Bed Description	134
5.3. Experiment Setup and Methodology	135
5.3.1. Network Performance Metrics	137
5.3.2. Scenarios	141
5.4. Results of Performance Evaluation	143
5.4.1. DAIM Communication Channel Results	143
5.4.2. Layer 2 Learning Switch Application Results	149
5.5. Build a Physical OpenFlow Test Lab Controlled by DAIM	154
5.5.1. Configuration Summary	155
5.5.2. Setup OpenFlow Switch and DAIM Controller on a Raspberry Pi	156
5.5.3. Basic Test	157
5.5.4. Preliminary Hardware Performance Results	158
5.6. Other Parameters That Can Affect the Performance Evaluation	160

III. Drawing Conclusions **163**

6. Conclusion and Future Work **165**

6.1. Research Propositions Validation	166
---	-----

6.2. Research Contributions and Findings	170
6.3. Research Limitations	172
6.4. On-going Work and Future Directions	173
Bibliography	175
A. DAIM Source Code for Data Analysis	185
A.1. Cross-controller Communications	187
B. Create OpenFlow Network with Multiple PCs	191
B.1. Configuration Summary	191
B.2. Assigning Static IP Address for Network Interfaces	192
B.3. Set Bridge IP Address for NOX Controller	192
B.4. NOX Controller Setup	193
B.5. Installing OpenVswitch on a Node	194
B.6. Installing OpenFlow Switching Reference System	196
B.7. NOX Controller Graphical User Interface (GUI)	196
B.8. Installing OpenFlow Wireshark Dissector	197
C. OpenFlow Laboratory with Mininet	199
C.1. Setting up Mininet Environment	199
C.2. Experimenting with Mininet	201
C.3. Running External Controllers	205
D. OpenFlow Setup in OMNeT++ INET Framework	207
D.1. Installing OMNeT++ 4.2	207
D.2. Configuring and Building OMNeT++	208
D.3. Verifying the Installation	208
D.4. Starting the IDE	208
D.5. Installing INET Framework 2.0	209
D.6. Installing OpenFlow Extension for the OMNeT++	209
D.7. Example of Simple OpenFlow testing in OMNeT++	210
E. Hardware for OpenFlow Test Lab	215

List of Figures

1.1. The Connections and Components of Two-Host Network Created by Mininet[57]	14
1.2. Mininet Emulation Software	15
1.3. Thesis Structure	17
2.1. Four Elements of Policy-Based Framework [112]	29
2.2. Centralised Network Paradigm [84]	31
2.3. SDN Evolution - Segregation of Control and Data Plane [97]	38
2.4. The Three-Tier Logical Layers of SDN[44]	39
2.5. Idealised OpenFlow Switch. A remote controller manages the Flow Table via the Secure Channel.	42
2.6. Flowchart Detailing Packet Flow Through OpenFlow Switch [33]	44
2.7. Components of a Flow Entry in a Flow Table	45
2.8. OpenFlow-enabled Switch with Flow Entries [116]	47
2.9. Components of a NOX-based network: OpenFlow (OF) switches, a server running a NOX controller process, and a centralised database containing the network view [47].	51
2.10. Example NOX-based network setup. Each switch has its own controller but network state is stored centrally [116].	53
2.11. Mapping of OpenFlow Network Protocol Layers	54
2.12. Self-X Functions [53]	68
2.13. Autonomic Computing Tree [102]	70
3.1. Comparison Between Traditional SNMP MIB and O:MIB [84]	75
3.2. Algorithms and Methods in O:MIB [29]	76
3.3. Self-Maintained Process [20]	77
3.4. Script Sample of Method Described O:XML Format for O:MIB [32]	79
3.5. Integration of Multi-Agent Framework with O:XML Implemented O:MIB [30]	80
3.6. The Mapping of Conventional Networks and SDN	82
3.7. DAIM Model Architecture as an Intelligent Computational Environment	86
3.8. DAIM Agent Owns a Flow Entry in the Flow Table	87
3.9. Flow Chart Detailing Packet Processing Within DAIM Model	89
4.1. DAIM Implementation Phase 1	97
4.2. DAIM Implementation Phase 2	98

4.3.	DAIM Implementation Phase 3	100
4.4.	DAIM Model Ecosystem	102
4.5.	Implemented OpenFlow Messages	103
4.6.	Unix Socket Connection Setup	105
4.7.	DAIM Storage Block of Memory (Object)	109
4.8.	Flowchart Detailing the Process of add_object ()	111
4.9.	Flowchart Detailing the Process of remove_object ()	112
4.10.	Flowchart Detailing the Process of free_list ()	113
4.11.	Packet Flow in an OpenFlow Switch Controlled by DAIM	116
4.12.	DAIM Model Integration with Mininet	119
4.13.	Simple Linear Topology Setup	121
4.14.	Ring Network Topology Setup	121
4.15.	Tree Network Topology Setup	122
4.16.	Fully Mesh Network Topology Setup	122
4.17.	Communication between two nodes in an OpenFlow network managed by DAIM.	124
4.18.	Screenshot of netcat UDP Chat Session	129
4.19.	Screenshot of VLC Video Streaming Session	130
4.20.	Screenshot of HTTP Web Server Session	132
5.1.	Scenario Used to Evaluate Mean RTT and Maximum TCP Bandwidth	141
5.2.	Scenario Used to Evaluate the Flow Setup Throughput and Latency	142
5.3.	Mean RTT DAIM Channel and NOX	144
5.4.	Mean RTT DAIM Channel and POX	145
5.5.	Number of Flow Requests Handled per Second	146
5.6.	Delay to Respond to Flow Requests	147
5.7.	TCP Bandwidth Utilisation Comparison	148
5.8.	Mean RTT Comparison	150
5.9.	Average Maximum Throughput Achieved with Different Number of MACs	151
5.10.	Flow Setup Latency Comparison	153
5.11.	Network Bandwidth Comparison	154
5.12.	Physical OpenFlow Test Lab Topology	155
A.1.	Controller Message from DAIM 1 to DAIM 2	189
A.2.	Controller Message from DAIM 2 to DAIM 1	189
B.1.	OpenFlow-Based SDN Lab Using OpenVswitch and Controlled by NOX via OpenFlow Protocol	191
B.2.	OpenFlow Dissector in Wireshark	198
C.1.	Enabling X11 Forwarding in PuTTY	200
C.2.	OpenFlow Laboratory Using Mininet	201

D.1. OpenFlow Mesh Topology with Spanning Tree Protocol	211
D.2. Measured RTT of TCP/IP vs. OpenFlow	213
E.1. A Small-Size OpenFlow Network	216
E.2. Raspberry Pi 2 Model B 1GB	216

List of Tables

2.1.	Characteristics of a Reliable Distributed System	33
2.2.	Details of Flow Headers (Twelve Tuples)	46
2.3.	Current Software Switch Implementations Compliant with the Open- Flow Standard	48
2.4.	Current Available Hardware Switches by Markets, Compliant with the OpenFlow Standard	48
2.5.	Current SDN Controller Implementations Compliant with the Open- Flow Standard	50
2.6.	A Comparison of NS-3, OMNeT++, Mininet and EstiNet [115] . . .	58
2.7.	DIFANE Wild-Card Rules	64
3.1.	Comparison of Normal and Candidate Processes	90
4.1.	DAIM Application Header Files	103
4.2.	API Dependencies Used to Implement DAIM Modules	104
4.3.	Public Functions with Associate Actions in the Object List Class . .	112
4.4.	OpenFlow Messages Handled by the Controller Module	114
5.1.	The Minimum Response Time	152
5.2.	SDN/OpenFlow Controllers: Code Extension	160

Nomenclature

AA	Autonomous Agent
ACNs	Autonomic Communication Networks
ACs	Autonomic Communications
ADSs	Autonomous Decentralised Systems
API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
BGP	Border Gateway Protocol
CLI	Command Line Interface
DAI	Distributed Artificial Intelligent
DAIM	Distributed Active Information Model
DHCP	Dynamic Host Configuration Protocol
DMI	Desktop Management Interface
DMTF	Distributed Management Task Force
DNS	Domain Name System
FCAPS	fault, configuration, accounting, performance, security
FTP	File Transfer Protocol
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
LLDP	Link Layer Discovery Protocol
LTE	Long Term Evolution
MANET	Mobile Ad hoc Network

MEs	Managed Elements
MIB	Management Information Base
NETCONF	Network Configuration Protocol
NFV	Network Functions Virtualisation
NGN	Next Generation Network
NIB	Network Information Base
NOS	Network Operating System
OFLOPS	OpenFlow Operations Per Second
OSCA	Operating System Communication Application
OSPF	Open Shortest Path First
OSS	Operations Support System
OVSDB	OpenvSwitch Database Management Protocol
QoS	Quality of Service
RESTful	Representational State Transfer
RNC	Radio Network Controllers
SDN	Software-Defined Networking
SID	Shared Information and Data Model
SLAs	Service Level Agreements
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
STP	Spanning Tree Protocol
TLS	Transport Layer Security
TMF	TeleManagement Forum
UML	Unified Modelling Language
VETH	Virtual Ethernet

VMs	Virtual Machines
WASNs	Wireless Ad-hoc Sensor Networks
WiMAX	Worldwide Interoperability for Microwave Access
WSN	Wireless Sensor Network
XACML	eXtended Access Control Markup Language
XML	Extensible Markup Language
XML-RPC	XML-encode Remote Procedure Call

Acknowledgements

I would like to acknowledge all the support and encouragement received during my PhD research. Firstly, I would like to express my deep gratitude to Professor Robin Braun, who has been my supervisor and very good friend. His valuable guidance through this research was a great source of support and encouragement and always made me go that extra mile to solve the various problems that lead to this work. I cherished the opportunity to watch and learn from his knowledge and experience. His frequent insights and patience with me are always appreciated.

I also thank my co-supervisor, Dr. Bruce Moulton, for supporting me throughout this work. During the course of this research, I also benefited greatly from interactions and technical discussions with other talented and warm-hearted members from the CRIN centre, including Dr. Zenon Chaczko and Dr. Abdallah Al Sabagh, to whom I wish to give sincere thanks. The technical conversations with them also helped me over the course of this project not only in terms of resolving quick technical difficulties but also with regards to lightening up.

It has been my privilege to work closely with Ameen Banjar, my research collaborator and best friend, I am gratefully thanking him for his invaluable contributions and innovative ideas towards this project. My special mention also goes to Md. Imam Hossain, for helping me with the solution to the problems in C/C++ programming, and setting up the test bed used for this research. I am very proud of what we have achieved together, thank you both.

I would like to thank Suan Dusit University, Bangkok, Thailand for providing international student academic research scholarship, and Emeritus Professor Tony Moon for nominating me for UTS International Research Scholarship (IRS) to support my PhD degree financially.

I am truly indebted to all my friends who have supported me over the last few years: Wael Alenazy, Jiajia Shi, Raniyah Wazirali, Lucia Gordon, Anup Kale, Denise Umuhoza, Shaher Slehat and Sanya Khruahong for their help in various ways. I have enjoyed many useful and happy chats with them. I have been very fortunate to have them around during my PhD study.

Last but by no means the least, I wish to give special thanks to my lovely family, Dr. Sawarng, Asst. Prof. Dr. Kanungnit and my sister Pitinut, for their immense support and all of the sacrifices that they have made on my behalf. My parents always gave me constant support and tried to provide me with the best education they can afford. They have been an important driving force to encourage me behind this PhD research.

I would like to express my sincere appreciation to my beloved wife, Duangporn, for her endless support, dedicated love, patience and understanding in every possible way. She always supported me in the moments when there was no one to answer my queries. Finally, I would like to thank my sons, Neptune and Neymar, for the happiness and cheers they have given me as their own way to support me.

Related Publications

The technical contributions and discussions in the thesis are mainly based on the following publications written by the author in which five of them are co-authored:

A. International Journal Publications:

[J1] Papatwibul, Pakawat, Ameen Banjar, Abdallah AL Sabbagh, and Robin Braun. "A Comparative Review: Accurate OpenFlow Simulation Tools for Prototyping." *Journal of Networks* 10, no. 5 (2015): 322-327.

[J2] Banjar Ameen, Pakawat Papatwibul, and Robin Braun. "DAIM: A Mechanism to Distribute Control Functions Within OpenFlow Switches." *Journal of Networks* 9.1 (2014): 1-9.

[J3] Banjar Ameen, Pakawat Papatwibul, Abdallah AL Sabbagh, and Robin Braun. "Using an ICN Approach to Support Multiple Controllers in OpenFlow." *International Journal of Electrical & Computer Sciences* 14, no. 2 (2014).

B. International Conference Publications:

[C1] Papatwibul, Pakawat, Ameen Banjar, and Robin Braun. "Using DAIM as a Reactive Interpreter for OpenFlow Networks to Enable Autonomic Functionality." *ACM SIGCOMM Computer Communication Review*. Vol. 43. No. 4. ACM, 2013.

[C2] Papatwibul, P., Jozi, B. and Braun, R. 2011, "Investigating O:MIB-Based Distributed Active Information Model (DAIM) for Autonomics", *International Conference on Information and Communication Technologies and Applications IIIS*, Orlando, Florida, USA, pp. 7-12.

[C3] Papatwibul, P., Sabbagh, A.A.L., Banjar, A. & Braun, R. 2012, "Distributed Systems in Next Generation Networks", *1st Australian Conference on the Applications of Systems Engineering ACASE'12*, p. 32.

[C4] Papatwibul, P., Banjar, A., Al Sabbagh, A., & Braun, R. (2013, October). "Developing an Application Based on OpenFlow to Enhance Mobile IP Networks". In *Local Computer Networks Workshops (LCN Workshops)*, 2013 IEEE 38th Conference on (pp. 936-940). IEEE.

[C5] Al Sabbagh, A., Papatwibul, P., Banjar, A., & Braun, R. (2013, October). "Optimization of the OpenFlow Controller in Wireless Environments for Enhancing Mobility". In *Local Computer Networks Workshops (LCN Workshops)*, 2013 IEEE 38th Conference on (pp. 930-935). IEEE

[C6] Banjar, A., Papatwibul, P., Braun, R., & Moulton, B. (2014, February). "Analysing the Performance of the OpenFlow Standard for Software-Defined Networking Using the OMNeT++ Network Simulator". In Computer Aided System Engineering (APCASE), 2014 Asia-Pacific Conference on (pp. 31-37). IEEE.

[C7] Papatwibul, P., Banjar, A. & Braun, R. 2014, "Performance Evaluation of TCP/IP vs. OpenFlow in INET Framework Using OMNeT++, and Implementation of Intelligent Computational Model to Provide Autonomous Behaviour." The Asian Conference on Technology, Information & Society 2014, The International Academic Forum (IAFOR) Osaka, Japan, pp. 43-56.

C. International Book Chapter Publications:

[B1] Papatwibul, Pakawat, Ameen Banjar, Abdallah AL Sabbagh, and Robin Braun. "An Intelligent Model for Distributed Systems in Next Generation Networks." In Advanced Methods and Applications in Computational Intelligence, pp. 315-334. Springer International Publishing, 2014.

[B2] Banjar, Ameen, Pakawat Papatwibul, and Robin Braun. "Comparison of TCP/IP Routing Versus OpenFlow Table and Implementation of Intelligent Computational Model to Provide Autonomous Behaviour." Computational Intelligence and Efficiency in Engineering Systems. Springer International Publishing, 2015. 121-142.