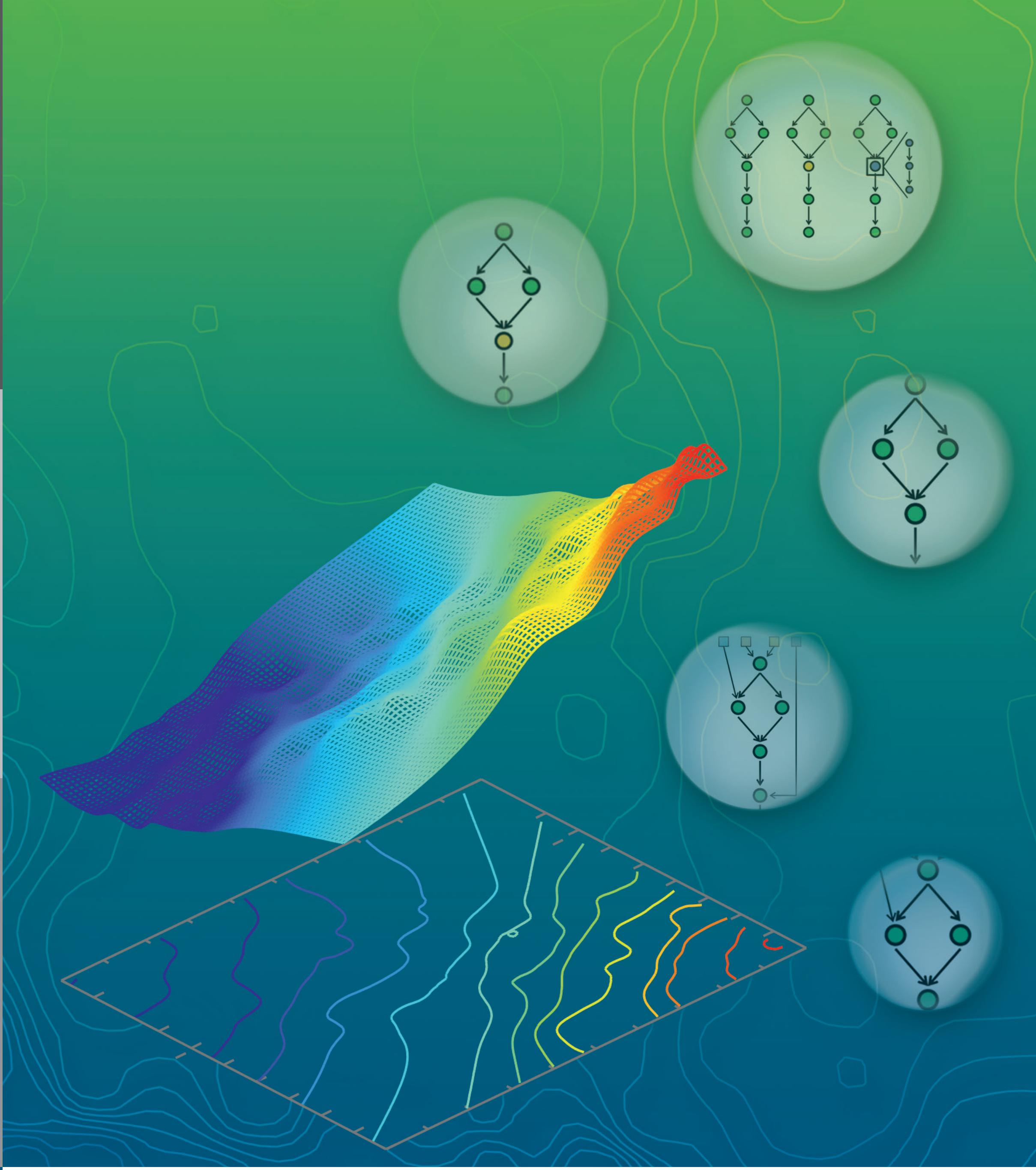


Automated Optimization Methods for Scientific Workflows in e-Science Infrastructures

Sonja Holl

JÜLICH
FORSCHUNGSZENTRUM

Schriften des Forschungszentrums Jülich

IAS Series Volume 24

Forschungszentrum Jülich GmbH
Institute for Advanced Simulation (IAS)
Jülich Supercomputing Centre (JSC)

Automated Optimization Methods for Scientific Workflows in e-Science Infrastructures

Sonja Holl

Schriften des Forschungszentrums Jülich

IAS Series Volume 24

ISSN 1868-8489 ISBN 978-3-89336-949-2

Bibliographic information published by the Deutsche Nationalbibliothek.
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at http://dnb.d-nb.de.

Publisher and Distributor:	Forschungszentrum Jülich GmbH Zentralbibliothek 52425 Jülich Phone +49 (0) 24 61 61-53 68 · Fax +49 (0) 24 61 61-61 03 e-mail: zb-publikation@fz-juelich.de Internet: http://www.fz-juelich.de/zb
Cover Design:	Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH
Printer:	Grafische Medien, Forschungszentrum Jülich GmbH
Copyright:	Forschungszentrum Jülich 2014

Schriften des Forschungszentrums Jülich
IAS Series Volume 24

D 5 (Diss., Bonn, Univ., 2014)

ISSN 1868-8489
ISBN 978-3-89336-949-2

Persistent Identifier: urn:nbn:de:0001-2014022000
Resolving URL: http://www.persistent-identifier.de/?link=610

Automated Optimization Methods for Scientific Workflows in e-Science Infrastructures

Dissertation
zur
Erlangung des Doktorgrades (Dr. rer. nat.)
der
Mathematisch-Naturwissenschaftlichen Fakultät
der
Rheinischen Friedrich-Wilhelms-Universität Bonn

vorgelegt von

Sonja Holl

aus
Mönchengladbach

Bonn, September 2013

Angefertigt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn

1. Gutachter: Prof. Dr. Martin Hofmann-Apitius

2. Gutachter: Prof. Dr. Heiko Schoof

Tag der Promotion: 27.01.2014

Erscheinungsjahr: 2014

IN DER DISSERTATION EINGEBUNDEN:

Zusammenfassung

Abstract

Scientific workflows have emerged as a key technology that assists scientists with the design, management, execution, sharing and reuse of *in silico* experiments. Workflow management systems simplify the management of scientific workflows by providing graphical interfaces for their development, monitoring and analysis. Nowadays, e-Science combines such workflow management systems with large-scale data and computing resources into complex research infrastructures. For instance, e-Science allows the conveyance of best practice research in collaborations by providing workflow repositories, which facilitate the sharing and reuse of scientific workflows. However, scientists are still faced with different limitations while reusing workflows. One of the most common challenges they meet is the need to select appropriate applications and their individual execution parameters. If scientists do not want to rely on default or experience-based parameters, the best-effort option is to test different workflow set-ups using either trial and error approaches or parameter sweeps. Both methods may be inefficient or time consuming respectively, especially when tuning a large number of parameters. Therefore, scientists require an effective and efficient mechanism that automatically tests different workflow set-ups in an intelligent way and will help them to improve their scientific results.

This thesis addresses the limitation described above by defining and implementing an approach for the optimization of scientific workflows. In the course of this work, scientists' needs are investigated and requirements are formulated resulting in an appropriate optimization concept. In a following step, this concept is prototypically implemented by extending a workflow management system with an optimization framework, including general mechanisms required to conduct workflow optimization. As optimization is an ongoing research topic, different algorithms are provided by pluggable extensions (plugins) that can be loosely coupled with the framework, resulting in a generic and quickly extendable system. In this thesis, an exemplary plugin is introduced which applies a Genetic Algorithm for parameter optimization. In order to accelerate and therefore make workflow optimization feasible at all, e-Science infrastructures are utilized for the parallel execution of scientific workflows. This is empowered by additional extensions enabling the execution of applications and workflows on distributed computing resources.

The actual implementation and therewith the general approach of workflow optimization is experimentally verified by four use cases in the life science domain. All workflows were significantly improved, which demonstrates the advantage of the proposed workflow optimization. Finally, a new collaboration-based approach is introduced that harnesses optimization provenance to make optimization faster and more robust in the future.

Acknowledgements

I would like to express my gratitude to all the people who supported me in any way during my PhD project. First and foremost, I would like to thank Prof. Martin Hofmann-Apitius for his advice and guidance though such a versatile research project. Visiting his research department was a wonderful working experience, especially collaborating with such a friendly and open minded group. I would like to thank Prof. Thomas Lippert and Daniel Mallmann for the opportunity to conduct my PhD research within the Federated Systems and Data group at the Jülich Supercomputing Center in the Forschungszentrum Jülich. It was a great workplace offering both an excellent technical infrastructure and social environment.

I am also very thankful to my supervisor Olav Zimmermann for his continuous support, fruitful discussions and numerous comments helping me sharpen my doctoral studies. Moreover, I thank Prof. Heiko Schoof to act as a very excited co-referee as well as further referees, Prof. Rainer Manthey and Prof. Diana Imhof.

I would like to acknowledge all my colleagues at the Jülich Supercomputing Center for providing technical background during the implementation, enormous efforts to create a stable and reliable infrastructure, proof-reading parts of this thesis, and offering a warm and cheerful office atmosphere in the coldest office in JSC. Additional thanks go to all the Studium Universale members offering friendly meetings as a welcome alternation to the PhD work.

A special thanks goes to Prof. Carole Goble for hosting me at the University of Manchester and her team, in person Dr. Khalid Belhajjame, Alan Williams, Stian Soiland-Reyes, Dr. Alexandra Nenadic and Dr. Katy Wolstencroft for providing excellent support and assistance during the implementation phase on extensions to the Taverna Workflow Management System and Research Objects.

Many thanks are due to my collaboration partners namely Prof. Magnus Palmblad, Dr. Yassene Mohammed, Daniel Garijo, Dr. Matthias Obst, Renato De Giovanni, Shweta

Bagewadi, and Dr. Philipp Senger for their fruitful discussions and great assistance with biological, medical or workflow provenance issues.

Finally, I am deeply grateful to my parents and grandparents as well as my two and '*a half*' brothers for being always there for me and supporting me in all my plans. I am indebted to Jen for his patience and never ending encouragement together with my flat mates, close friends, and people I met by carpooling for making my time in Jülich and Aachen a funny and relaxing one. Thanks for your open ears and hearts in all situations, continuous support as well as proof-reading parts of this thesis. Thank you all for walking along side with me this long and finally successful path.

Sonja Holl
January 2014

Contents

List of Figures ix

List of Tables xi

List of Abbreviations xiii

List of Publications xv

1 Introduction 1

1.1 Scientific Workflows in e-Science 1

1.2 Challenges for Scientists Using Life Science Workflows 3

1.3 Goals of the Thesis 5

2 Concept Development for State-of-the-Art Workflow Optimization 9

2.1 General Aspects of Scientific Workflows 9

2.1.1 Scientific Workflows 9

2.1.2 Scientific Workflow Management Systems 12

2.1.3 e-Science Collaborations 12

2.2 General Aspects of Optimization and Learning 14

2.2.1 Mathematical Background and Notations 14

2.2.2 Different Optimization Algorithms 15

2.2.3 Design Optimization Frameworks 17

2.3 State-of-the-Art Scientific Workflow Optimization 18

2.3.1 Runtime Performance Optimization 18

2.3.2 Output Performance Optimization 19

2.3.3 Other Concepts of Workflow Modification 20

2.4 A Concept for Scientific Workflow Optimization 21

3 Enabling Parallel Execution in Scientific Workflow Management Systems 27

3.1 Investigation of Scientific Workflow Management Systems in e-Science . 28

3.2 Extension of a Workflow Management System 32

3.2.1 The Taverna Workflow Management System 33

3.2.2 UNICORE Middleware 35

3.2.3 Architecture of the Grid Plugin 35

3.2.4 Development of the Grid Plugin 36

3.2.5 Enhanced Parallel Application Execution 38

3.3 Evaluation by Life Science Use Cases 39

3.4 Discussion 43

3.5 Conclusion 44

4 A Framework for Scientific Workflow Optimization 47

4.1 The Approach of Scientific Workflow Optimization 48

4.1.1 A New Optimization Phase in the Scientific Workflow Life Cycle 48

4.1.2 Investigation of Different Optimization Levels 51

4.1.3 Definition of the Optimization Target 54

4.2 The Usability Compliance of Workflow Optimization 55

4.3 The Taverna Optimization Framework 56

4.4 Enabling Optimization on Distributed Computing Infrastructures 60

4.4.1 Three Tier Execution Architecture 61

4.4.2 Implementation of Parallel Workflow Execution 62

4.4.3 Parallel Optimization Use Case 64

4.5 Discussion 64

4.6 Conclusion 65

5 Optimization Techniques for Scientific Workflow Optimization 67

5.1 Optimization Techniques for Scientific Workflow Parameters 67

5.1.1 Genetic Algorithms 70

5.1.2 A Genetic Algorithm for Scientific Workflows 71

5.2 The Parameter Optimization Plugin 72

5.2.1 Development of the Parameter Optimization Plugin 73

5.2.2 Discussion 76

5.3 Evaluation of the Parameter Optimization Plugin 76

5.3.1 Proteomics Workflows 77

5.3.2 Ecological Niche Modeling Workflows 84
5.3.3 Biomarker Identification Workflows 89
5.3.4 Protein Structure Similarity Workflows 92
5.3.5 Discussion 94
5.4 Simulation of Workflow Structure Optimization 96
5.4.1 The Component Level 96
5.4.2 Discussion 99
5.4.3 The Topology Level 100
5.4.4 Discussion 102
5.5 Conclusion 102

6 Discussion: Scientific Workflow Optimization in e-Science 105
6.1 Examination of Scientific Workflow Optimization 106
6.1.1 General Aspects of Optimization 106
6.1.2 Addressing Optimization Complexity 111
6.2 Provenance-based Optimization 115

7 Conclusion 123
7.1 Summary of the Work 123
7.2 Future Work 126

A Additional Figures and Descriptions 131

Bibliography 149

List of Figures

2.1 The main workflow structures 10

2.2 A workflow in the context of optimization 22

2.3 Three different requirements to investigate workflow optimization 24

3.1 Three tier concept for workflow optimization. First approach: acceleration of compute-intensive applications 27

3.2 The Taverna Workbench 34

3.3 Taverna parallel execution 34

3.4 System architecture of the UNICORE-Taverna plugin 37

3.5 Sweep jobs in Taverna 40

3.6 The X!Tandem workflow 41

3.7 Concept of tandem mass spectrometry 42

4.1 Three tier concept for workflow optimization. Second goal: generic and automated approach to be multipurpose extensible 47

4.2 Extended scientific workflow life cycle 49

4.3 Three defined levels for workflow optimization 52

4.4 The Taverna optimization perspective 56

4.5 Architecture of the new framework 57

4.6 Taverna dispatch stack 58

4.7 Control flow of the new optimize layer 59

4.8 Three tier execution architecture 61

4.9 The new security propagation mechanism 63

4.10 Client workload of two different scenarios 64

5.1 Three tier concept for workflow optimization. Third goal: addressing the non-linear optimization problem 68

5.2 A workflow to Genetic Algorithm encoding mechanism 70

5.3 Abstract programming interface in detail 73
5.4 Plugin control flow 74
5.5 Optimization process screenshot 75
5.6 First iteration of Proteomics workflow 79
5.7 Plot of the Proteomics workflow optimization 81
5.8 General principle of Ecological Niche Modeling 84
5.9 Abstract Ecological Niche Modeling workflow 85
5.10 Biomarker identification workflow 90
5.11 Support vector regression workflow 93
5.12 Workflow for retention time prediction optimization 98
5.13 Root mean square deviation values 99
5.14 Abstract BLAST workflow 102

6.1 Optimization Research Object Ontology 118

7.1 Three tier concept and implementation for workflow optimization 124

A.1 Sequence logo of hybrid *E. coli* 131
A.2 Peptide identifications 132
A.3 Sequence logo of *E. coli* 133
A.4 ENM workflow with SVM 136
A.5 ENM workflow with Maxent. 137
A.6 *Crassostrea gigas*, SVM. 138
A.7 *Crassostrea gigas*, SVM comparison 139
A.8 *Prorocentrum minimum*, SVM 140
A.9 *Prorocentrum minimum*, SVM comparison 141
A.10 Diagram of the RO-Opt Algorithm 142
A.11 Diagram of the RO-Opt Fitness 143
A.12 Diagram of the RO-Opt Optimization Run 144
A.13 Diagram of the RO-Opt Search Space 145
A.14 Optimization Provenance Ontology 147
A.15 BioVel Optimization Provenance 148

List of Tables

3.1 Evaluation of common scientific workflow management systems 31
3.2 Description of UNICORE services 36
3.3 Submission via the conventional submission mechanism and the developed sweep generator 43

5.1 Results of the optimization of X!Tandem 80
5.2 Fitness values of interchanged MME+ and MME- values 82
5.3 Results of the optimization of X!Tandem including 4 different parameters 83
5.4 Results for default values and optimization of SVM algorithm 87
5.5 Results for default values and optimization of Maxent algorithm 88
5.6 Comparison of the intelligent feature ranking optimization results 91

List of Abbreviations

ACO	Ant Colony Algorithm
API	Application Programming Interface
AUC	Area Under the Curve
BPEL	Business Process Execution Language
DAG	Directed Acyclic Graph
DEISA	Distributed European Infrastructure for Supercomputing Applications
DN	Distinguished Name
EA	Evolutionary Algorithm
EFS	Ensemble Feature Selection
ENM	Ecological Niche Modeling
FDR	False Discovery Rate
GA	Genetic Algorithm
GUI	Graphical User Interface
HAB	Harmful Algae Bloom
HPC	High-Performance Computing
HTC	High-Throughput Computing
JERM	Just Enough Results Model
MIAME	Minimum Information About a Microarray Experiment
MIM	Minimum Information Model
MME	Mass Measurement Error
MO	Multi-Objective Optimization
MOEA	Multi-Objective Evolutionary Algorithm
PSM	Peptide Spectrum Match
PSO	Particle Swarm Optimization
RDF	Resource Description Framework

RFE	Recursive Feature Elimination
RO	Research Object
ROC	Receiver-Operating Characteristic
SA	Simulated Annealing
SPI	Service Provider Interface
SVM	Support Vector Machine
SVR	Support Vector Regression
SWMS	Scientific Workflow Management System
TPP	Trans-Proteomic Pipeline
VRE	Virtual Research Environment
XML	Extensible Markup Language
XSEDE	eXtreme Science and Engineering Discovery Environment

List of Publications

1. Sonja Holl, Yassene Mohammed, André M. Deelder, Olav Zimmermann, and Magnus Palmblad, „Optimized Scientific Workflows for Improved Peptide and Protein Identification“, *Molecular & Cellular Proteomics*, 2013, submitted: 4/7/2013

2. Sonja Holl, Daniel Garijo, Khalid Belhajjame, Olav Zimmermann, Renato De Giovanni, Matthias Obst, and Carole Goble, „On Specifying and Sharing Scientific Workflow Optimization Results Using Research Objects“, *The 8th Workshop on Workflows in Support of Large-Scale Science*, to be published, IEEE, 2013

3. Sonja Holl, Olav Zimmermann, Magnus Palmblad, Yassene Mohammed, and Martin Hofmann-Apitius, „A New Optimization Phase for Scientific Workflow Management Systems“, *Future Generation Computer Systems*, 2013, to be published

4. Sonja Holl, Mohammad Shahbaz Memon, Morris Riedel, Yassene Mohammed, Magnus Palmblad, and Andrew Grimshaw, „Enhanced Resource Management enabling Standard Parameter Sweep Jobs for Scientific Applications“, *9th International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems*, to be published, IEEE, 2013

5. Shahbaz Memon, Sonja Holl, Morris Riedel, Bernd Schuller, and Andrew Grimshaw, „Enhancing the performance of workflow execution in e-Science environments by using the standards based Parameter Sweep Model“, *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery*, XSEDE ’13, ACM, 2013, 56:1–56:7

6. Sonja Holl, Olav Zimmermann, and Martin Hofmann-Apitius, „A new optimization phase for scientific workflow management systems“, *2012 IEEE 8th International Conference on E-Science (e-Science)*, IEEE, 2012, pages 1–8

7. Sonja Holl, Olav Zimmermann, Bastian Demuth, Bernd Schuller, and Martin Hofmann-Apitius, „Secure Multi-Level Parallel Execution of Scientific Workflows on HPC Grid Resources by Combining Taverna and UNICORE Services“, *Proceedings of UNICORE Summit 2012*, Schriften des Forschungszentrums Jülich, IAS Series 15, Forschungszentrum Jülich, 2012, pages 27–34

8. Sonja Holl, Olav Zimmermann, and Martin Hofmann-Apitius, „A UNICORE Plugin for HPC-Enabled Scientific Workflows in Taverna 2.2“, *2011 IEEE World Congress on Services*, IEEE, 2011, pages 220–223

9. Bastian Demuth, Sonja Holl, and Bernd Schuller, „Extended Execution Support for Scientific Applications in Grid Environments“, *Proceedings of UNICORE Summit 2010*, Schriften des Forschungszentrums Jülich, IAS Series 5, Forschungszentrum Jülich, 2010, pages 61–67

Chapter 1

Introduction

1.1 Scientific Workflows in e-Science

In recent years, scientists have been enhancing their research activities by using computing technologies [Hey2009; Bell2009]. Along with theory and laboratory experiments, computer simulations and data analyses are used to test and validate scientific hypotheses in order to gain new insights in scientific phenomena. Especially within the life sciences difficult, expensive or dangerous tests are implemented by so-called *in silico* experiments – computer-based experiments. For example, new drug candidates can be tested and presorted by using high-throughput virtual screening methods before validating them in the laboratory. On that account, the interaction and binding of small molecules to target proteins is predicted in order to evaluate the inhibition or activation of specific functions. This reduces the number of required laboratory tests and in this way drug discovery is accelerated by adding *in silico* models instead of using traditional *in vivo* and *in vitro* experimentation only.

In silico data analyses and simulations are accomplished by composing different combinations of software applications, which use data and computing resources. Not long ago, these computing studies were implemented by executing shell scripts or other scripting languages [McPhillips2009; Ludascher2009b]. Many scientists no longer preferred this type of processing due to the fact that scripts are difficult to program and reuse, especially when using different computing or data resources [Wolstencro2009; Jimenez2013].

As a result, scientific workflows [Deelman2006b; Gil2007] emerged, which describe the different compositions of existing algorithmic building blocks that formulate large *in silico* experiments. Scientific workflows provide a high level abstraction of the linkage of individual processing steps. They facilitate not only the automation and comprehension but also the editing. Furthermore, their comprehensibility makes them suitable for dissemination, reuse and reproducibility of best practice analysis recipes. Scientific workflows were quickly adopted by life science communities [Taylor2006a] as they can record the knowledge discovery process [Gil2008; Ludascher2009a]. Scientific workflows facilitate the reuse and reproducibility of scientific experiments with minimal effort. Scientists can perform identical or similar workflows without reinventing the entire experiment. Platforms that allow sharing scientific workflows by collaborative means are workflow repositories [Stoyanovic2010]. These scientific workflow repositories enable scientists to share scientific knowledge and facilitate distributed collaborations. An example is the myExperiment platform [Goble2010], which contains a highly popular workflow repository. As a consequence, scientific workflows play a substantial role for domain scientists as a graphical alternative to scripting and programming [De Roure2009c]. Furthermore, they are highly important for reuse and reproducibility purposes and also provide an instrument for collaborative work and knowledge sharing.

Scientific workflows also contain several data-intensive analyses and compute-intensive applications. These applications can exceed the power of desktop clients or laptops, which makes it necessary to utilize distributed data and computing resources [Mehta2012], especially including High-Performance Computing (HPC) and High-Throughput Computing (HTC) resources. The utilization of remote resources is provided by Scientific Workflow Management Systems (SWMSs), which are a further step to support domain scientists [Deelman2009]. Many SWMSs have been developed for the life sciences [Achilleos2012], for example Taverna, Kepler, Knime or PipelinePilot, all providing similar functionalities but diverse technical details. They are software packages, enabling the graphical design, one-click execution and monitoring of local or distributed scientific workflows.

In recent years, the combination of both, collaboration techniques and access to distributed computing and data resources required for scientific workflows, have evolved as the term *'enhanced science'* (e-Science) [Hey2005]. E-Science was originally shaped in 1999 by John Taylor, Director of the Research Councils in the UK as the following: *'e-Science is about global collaboration in key areas of science and the next generation infrastructure that will enable it'* [NESC1999]. So called e-Science infrastructures comprise different services that enable seamless and secure inter-disciplinary collaborations (e.g. sharing of workflows) as well as provisioning of computing power [Taylor2006a] that can be employed by compute- or data-intensive life science workflows [Hey2003; Mehta2012]. Access to those infrastructures and especially distributed computing resources is granted for example by so-called Grid and Cloud middleware [Lin2011]. Middlewares supply an abstract layer above computing and data resources to provide services that allow easy and seamless access to distributed computing resources. *In silico* experiments designed as scientific workflows can be scaled up and accelerated by utilizing distributed computing and data resources in SWMSs.

With rising popularity of workflow-based e-Science infrastructures, best practice workflows can be shared, disseminated, searched, reused, validated and manipulated [Goble2007], just like within the myExperiment platform. However, even if a suitable workflow is found or a new workflow is designed from scratch, it is still required to set up the workflow regarding the specific problem at hand to improve the scientific results. Consequently, one might have to exchange applications and select configuration parameter and data inputs. Regardless of the user's prior computer experience level, challenges arise with trial and error or parameter sweeps providing the only option to evaluate different workflow set-ups for mining possible solutions.

The next sections examine these challenges and scientists' needs to investigate an alternative approach improving the procedure of setting up scientific workflows.

1.2 Challenges for Scientists Using Life Science Workflows

Scientific workflows have emerged as a useful instrument to comprehensively design and share the best practices and create reproducible scientific experiments [Littauer2012].

The workflows are designed from scratch by scientists or reused as a template from a workflow repository. After the design or searching phase, a scientist would end up with a workflow almost ready for execution. Nevertheless, as the workflow set-up depends on the used data and scope of experiment, the used components and their parameter settings may require adaptation according to the problem at hand and personal preferences to improve workflow results. Due to today's abundance of life science applications [Tiwari2007] and the large number of available parameters provided by these components [Kulyk2008; Kulyk2006], workflow adaption constitutes a significant difficulty. Kulyk et al. [Kulyk2008] stated that parameters enhances the flexibility of applications but make them more complex. This implies that improving a scientific workflow result may be challenging for both life science groups, experimentally affine domain scientists and computer affine bioinformaticians.

Assuming that the applied workflow and applications contain default parameters, domain scientists would typically keep these default values as they do not know how the programs exactly work and how parameter changes may influence the result [Kulyk2008]. If they do not want to rely on default parameters, they reuse their own or third party generated workflow parameters and test them manually [Kulyk2008]. This trial and error process may become time consuming and ineffective. Due to the rising complexity of scientific workflows [Juve2012a] it may also become difficult to manage and overlook the individual workflow set-ups.

A recent analysis [Starlinger2012] also found that scientists develop and share designed workflows, but tend to almost exclusively reuse their own scientific workflows, instead of adapting workflows from other authors. Due to the fact that current repositories are still not large enough to provide a high number of relevant workflows, it is nevertheless justified to question how the reuse of workflows might be increased and how shared workflows can be turned into best practice workflows for a broad range of scientists in the future. Currently, domain scientists use the default values, rather than developing or manually adapting solutions themselves [CohenBoula2011].

Whereas trial and error might become time consuming, parameter sweeps are a commonly used method by bioinformaticians. Many publications aim at supporting more appropriate solutions to speed up or ease the use of parameter sweeps (examples are [Oliveira2011], [Chirigati2012], or [Smanchat2013]). Bioinformaticians do not hesitate to use advanced methods as they design and program their own tools, which leads to the fact that they know how many programs work [Wassink2010]. However, with an increasing number of parameters, it is difficult to gather interrelations and dependencies of applications and parameters while using parameter sweeps. Additionally, the number of possible workflow set-ups may increase quickly and become computationally intensive. Certainly, parameter sweeps can be performed on a sample data set only to reduce the execution time. But nonetheless an intelligent search method could sample the search space more effective than a parameter sweep method can.

Many bioinformaticians also prefer the usage of the command line [Kulyk2008] instead of using a SWMS. The command line enables easier customization of parameter settings and, furthermore, automated scripts can be performed to obtain optimized parameters.

However, domain scientists may not want to reuse these scripted workflows and methods, as scripts are difficult to reuse due to the missing abstraction of composition and distributed resource [Wolstencro2009]. In turn, providing these advanced methods within SWMSs would require extended control and larger changes or extensions on SWMSs. Therefore, bioinformaticians are missing a platform to provide these improved methods without much effort and knowledge about the specific SWMS to enable a customized optimization of workflow settings.

Summarizing, the challenges of domain scientists and bioinformaticians in utilizing scientific workflows are:

- Choices for workflow structure and parameters may affect the overall scientific result.
- Workflows that are either developed from scratch or used as best practice require manual set-up and adaption to relate to the scope of experiment.
- The number of available applications and parameters to be set is large.
- Trial and error parameter determination is time consuming and inefficient.
- Parameter sweep explorations expand fast with increasing number of parameters.
- Advanced optimization scripts exist but require effort to be included into SWMSs.

These challenges provide the motivation to investigate an automated approach that provides a generic method and supports the optimization of scientific workflows to foster the effective utilization of scientific workflows by domain scientists and bioinformaticians alike.

1.3 Goals of the Thesis

The previous section identified that both, domain scientists and bioinformaticians, are faced with various challenges during workflow utilization and therefore an automated and generic method for workflow optimization is desired. In order to test the feasibility of a general approach for scientific workflow optimization, this thesis examines the question and eventually presents a novel approach of how to assist scientists in the finding of useful workflow set-ups to obtain optimal scientific results. The approach should assist scientists in the process of workflow adaption in a simplified, time efficient and intelligent manner.

For a broad acceptance and collaborative use of workflow optimization, the general approach should be coupled with the commonly used workflow tools, i.e., SWMSs, available within e-Science infrastructures and should also consider different optimization targets.

As the optimization problems themselves may be as distinct as the diverse scientific workflows and their scopes, the use of different optimization algorithms may be required. Therefore it has to be examined which optimization algorithms are available and appropriate within the context of workflows. These algorithms may also target different levels of the workflow description, for instance parameter optimization or other workflow levels. As algorithms and levels could become obsolete at some point and researchers also want to have novel techniques available, it will not only be required to identify possible optimization objectives but moreover approach a technique that is able to subsequently incorporate novel optimizations. These methods should be easily insertable with less effort by bioinformaticians.

Similar to the time efficient methods developed for parameter sweeps [Oliveira2011; Chirigati2012; Smanchat2013], a time efficient workflow optimization technique has to be taken into account. In doing so, easily accessible resources and consequently those, which are already available within e-Science environments should be used.

In the course of this thesis, these challenges will be addressed by an investigation and development of a workflow optimization approach embedded in a user-friendly research environment to be used by experts and non-experts alike. This thesis will explore a time efficient and extensible approach as well as investigate possible methods and targets to pursue workflow optimization. These approaches will be generally assessed and therefore this thesis makes the following contributions:

- Requirement analysis and development of a concept for scientific workflow optimization.
- Design and implementation of an automated and generic method for workflow optimization.
- Introduction of an optimization plugin employed on workflow parameters.
- Provisioning of a collaboration-based approach to harness optimization provenance.

In order to experimentally verify these contributions made to target workflow optimization, four use cases were implemented. They tackle issues from the following life science domains:

- Proteomics
  A workflow for protein identification via tandem mass spectrometry. The workflow matches protein fragment weights from tandem mass spectrometry to a database for identification and evaluates the findings. This workflow was developed by the Leiden University Medical Center.

- Ecology
  Ecological Niche Modeling workflow to perform species adaptation to environmental changes, developed within the BioVel Project. The workflow takes recent occurrences of a species and environmental layers into account, in order to create a model and predict the species' resistance level in the present or/and in the future (e.g. 2050).

- Medical Informatics
  Ranking of genes for intelligent feature selection. The workflow generates a ranked list of genes by performing a recursive feature elimination or ensemble feature selection by using several iterations of a support vector machine. This workflow was developed by the Fraunhofer Institute for Algorithms and Scientific Computing (SCAI).

- Structural Bioinformatics
  Estimating the local structure similarity of a protein segment via Support Vector Regression (SVR). The workflow trains a local protein structure predictor that estimates the structural similarity of a protein segment, where only the sequence profile is known, to a reference structure. This workflow was developed by the Jülich Supercomputing Center, Forschungszentrum Jülich.

Based on the obtained results and experiences, the general approach will be discussed and a novel strategy on how to address identified issues in the future will be investigated.

This thesis is structured as follows: Chapter 2 starts with the basic introduction of scientific workflows, SWMSs, general aspects of optimization algorithms and related work. This is followed by the design for a workflow optimization approach from which the implementations within this thesis are derived. Chapter 3 consists of two parts: the first section will investigate different SWMSs, and the second section addresses the extension of the selected SWMS to support parallel execution of applications to augment execution performance for optimization. In Chapter 4 different levels of workflow optimization will be discussed in combination with the development of a new scientific workflow life cycle phase for optimization. Additionally, this phase will be designed and implemented as a novel generic and automated optimization framework. In Chapter 5, an example plugin for parameter optimization will be developed. This plugin will extend the proposed framework and four use cases will be tested to evaluate this thesis approach. A detailed discussion of the results will follow in Chapter 6 concluding with a novel approach for provenance-based optimization to overcome identified issues. Finally, Chapter 7 summarizes the thesis research and outlines future work.

Chapter 2

Concept Development for State-of-the-Art Workflow Optimization

This thesis develops an approach for workflow optimization in order to ease the finding of a best practice scientific workflow set-up. Before describing the investigation of a concept for workflow optimization in Section 2.4, the first three sections introduce the different topics to the reader. The first section provides an introduction on scientific workflows, SWMSs as well as their collaborative usage within e-Science infrastructures. A basic overview of optimization algorithms is given in the second section. Afterwards, related work is presented. Finally, the fourth section conveys the approach to design a concept for workflow optimization, based on the lessons learned from the previous sections.

2.1 General Aspects of Scientific Workflows

2.1.1 Scientific Workflows

Workflows in general have been standardized by the Workflow Management Coalition in 1994 [WMC2013]. Based on these building blocks scientific workflows [Singh1996] and business workflows [Aalst2004] evolved. As the scientific community had their own specific demands on workflows, both were separated at an early stage. This was already highlighted by Singh and Vouk [Singh1996] in the middle of 1996. Nevertheless, scientific workflows took a significant upturn only in the last decade [Ludascher2005; Deelman2006b; Fox2006; Barker2008] and still remain as a topic of research especially in the life sciences [Juve2012a; Achilleos2012].

Scientific workflows are a structured aggregation of computer-based components and precisely define their execution and data flow to automate a scientific experiment. Each component acts as an algorithmic fragment, producing, converting, modifying or consuming data by receiving data from various inputs and conveying data through outputs. Data is being processed by components in a pipeline until all have finished their scientific task and output one or several results. The type of a component can be of different nature, such as a software program, a Web service, a local shell script, a binary executable, a Java program, data transformation or a Grid job [Wassink2009b]. Components are connected by linking inputs and outputs as a data or execution link and thus defining the workflow composition. A workflow composition, typically represented as Directed Acyclic Graph (DAG) [Deelman2009], clearly defines the design of a scientific experiment by offering a simple visualization of a complex processing model. Figure 2.1 shows an abstract view of an exemplary workflow. The workflow is data-centric, and therefore the output from one or several parallel tasks serves as the input for the subsequent task. The main characteristics of a scientific workflow are:

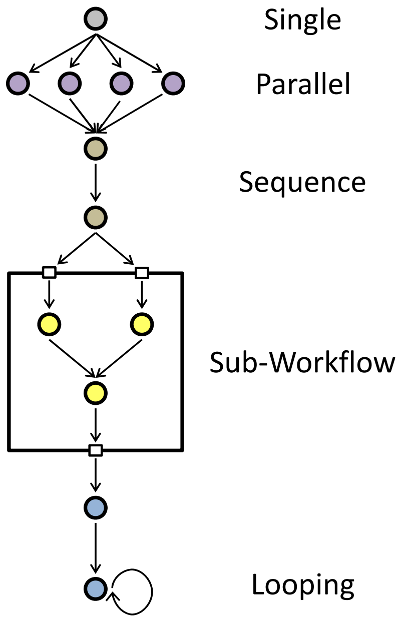


Figure 2.1: The main workflow structures: single task, parallel execution, sequence of tasks, sub-workflows of several tasks, and looping.

Single

A single component is executed, representing an algorithmic or functional task.

Sequence

Two or more components are executed in a pipeline, whereas exactly one task is followed by one other task. The later task is fed by the output of the previous task.

Parallel

Several single components are executed simultaneously usually in different threads. Parallel execution can be data parallel, where each component performs the same task with differing data, or parameter parallel, where each component performs the execution on the same data with different input parameters. Additionally, they can be executed in task parallel, where each sub-task executes a piece of a main task.

Typically, a preceding component is used for data separation or distribution and a subsequent component for data aggregation or merging.

Sub-Workflow

A sub-workflow defines a set of components connected in any of the defined characteristics. Sub-workflows can be seen as a group of tasks, typically aiming to perform a specific purpose. Sub-workflows can usually be out-zoomed (minimized) in the entire workflow to a single component.

Looping

A component can be executed in a loop. It is iteratively executed with one or several varying parameter or data inputs until a specific case comes true of false. (Loops are only simulated by the underlying workflow language and do not exist in the graph.)

For a detailed view of different structures and characteristics of scientific workflows, please refer to [Juve2012a] and for a general overview of workflows to [Aalst2003].

Scientific workflows may have different states during their evolution. In this thesis, four definitions will be fixed for a better understanding of the different workflow states:

Workflow template

A workflow template is an abstract draft of an *in silico* experiment. A template has abstract definitions of applications and their used data types. It holds no further information about the execution and resources.

Workflow specification

A workflow specification is a concrete scientific experiment recipe. Explicit applications are selected, the inputs are specified for data types and the resource demands are set.

Workflow instance

An instance is a specific instantiation of a workflow. All parameters are set, input data is processed and the execution will be performed on a specific target resource.

Workflow run

A workflow instance that has already been executed.

2.1.2 Scientific Workflow Management Systems

SWMSs are the method of choice to conduct scientific workflows [Curcin2008]. In the last decade, several SWMSs emerged [Deelman2009] all aiming at facilitating the design, composition, execution, monitoring and management of scientific workflows. This is achieved by automation and simplification so that scientists can focus on their research and are not required to concentrate on the technical details of data acquisition, component execution and resource allocation. A SWMS is in charge of data stage-in and stage-out, scheduling computational tasks and managing their dependencies. Data stage-in and stage-out describes the (transfer) process of loading and saving data on both the local system and a distributed resource environment. SWMSs partly vary in their properties, such as different workflow languages or graphical user interface concepts. Access to distributed computing resources is supported by many workflow systems whereas the implementations not only vary in the supported e-Science infrastructure but also in the supported Grid or Cloud middleware. More information and a precise distinction of SWMSs will be given in Chapter 3.

2.1.3 e-Science Collaborations

With the Web 2.0 [OReilly2009] community collaboration and social networking techniques, i.e., Virtual Research Environments (VREs), have been increasingly applied to scientific research. Collaboration, which means sharing, reuse and networking, has become an essential progress in science [Marx2012] where researchers want to investigate questions together that they cannot perform alone [Olson2008]. Especially scientific workflows received a lot of attention by the e-Science community due to their ability to comprise research processes and represent knowledge artifacts. The myExperiment project was launched in 2007 [De Roure2009b] and has grown to be the largest public repository of scientific workflows [De Roure2010]. myExperiment is a Web 2.0 based e-Science infrastructure to enable collaborations and sharing of knowledge within the life science community [De Roure2009d]. It comprises a social networking environment with a workflow repository in the background. Workflows stored in these repositories can be tagged, rated and grouped to be searchable and reusable by third party research groups. Scientific workflows from many domains can be found on myExperiment (at the time of writing over 2500 workflows [UoM2007]) originating from different workflow types and domains, such as bioinformatics, physics, chemistry or geoscience. Initially, its aim was at supporting bioinformaticians as a part of the myGrid project [UoM2008a] and the SWMS Taverna [Missier2010; UoM2009]. Thus, the life sciences have still a higher representation [Achilleos2012] than other research domains.

Within the last years, a significant effort has been made to develop myExperiment to a 3^{rd} generation VRE [De Roure2010]. Third generation VREs are different from the second generation with regard to the shared and reused objects. Whereas workflows were the main object itself back then, research is getting increasingly data-driven and so-called *packs* are gaining more attention nowadays. Packs are used within myExperiment and aggregate all objects associated with a specific experiment, for instance workflows, input and output data or publications. These aggregations show the process of how next generation research is going to take place. This new type of research is referred to as *linked data* [Bizer2009]. Linked data describes the process of publishing and connecting structured data on the Web. To support linked data in myExperiment [De Roure2010], a separate server hosting and publishing all myExperiment files as Resource Description Framework (RDF) structured data according to the myExperiment ontology was created firstly. Second, a SPARQL (SPARQL Protocol and RDF Query Language) Web service endpoint was created to allow the querying and retrieving of any type of myExperiment entity as an RDF description. These RDF entities cannot only be queried by users but also by third party linked data services, such as the public voiD (Vocabulary of InterLinked Datasets) store [Alexander2011].

The 3^{rd} generation data centric VREs are also taking the representation, storing and reusing of provenance into account. The storing of provenance data of scientific experiments plays an important role for the reusability and reproducibility [Achilleos2012]. Provenance within SWMSs can be applied at different levels whereas the typical workflow provenance includes workflow runs, specifications, and execution products. Each SWMS has its own mechanism and language to store and represent provenance; recent developments aim to harmonize these [De Roure2009a; Belhajjame2012a]. Provenance aggregates information about a workflow, including specifications, execution traces or data items. Thus, these artifacts hold details for a subsequent understanding like the workflow structure and experiment purpose. In an ideal scenario, users run workflows in workflow management systems or portals and these technologies store workflow provenance. Other scientists in turn can search and reuse the individual provenance objects to produce their own scientific results.

2.2 General Aspects of Optimization and Learning

Scientific workflow optimization is formulated as an optimization problem in this thesis. Similar to real world problems such as vehicle routing or managing the energy consumption, workflow optimization can be defined as the maximization or minimization of specific subjects regarding concrete constraints. This section gives the reader a basis to optimization algorithms used in science and engineering to provide general knowledge regarding workflow optimization.

2.2.1 Mathematical Background and Notations

The general process of mathematical optimization of a problem can be described as the systematical comparison of possible parameter sets and the finding of a not yet known optimal (i.e., the *best*) parameter set for a model. An optimization is focused at the maximization or minimization of so-called *objective functions* (or *fitness functions*) regarding various constraints. Within real-world problems, the typical goal is to find the best possible solution in a reasonable amount of time.

The model, which is the scope of the optimization, may be of various kinds and can range from design modeling such as aircraft wings [Sobieszczа1997] over power systems [AlRashidi2009] to drug discovery [Nicolaou2013]. Accordingly, objective functions differ in their type and scope, such as the drag and structural weight, the minimization of generator cost or the pharmacokinetic properties and synthetic accessibility, respectively.

The search space is spanned by the decision variables, while the solution space is defined by the objective functions and their constraints. The goal is to find an n dimensional vector of values in the search space, which corresponds to a global optimum in the solution space.

Optimization problems [Nocedal2006] can be classified into *single-objective optimiza-*

tion – only one fitness function is optimized – and *multi-objective optimization problems* – many fitness functions are optimized. Both cases are also called *single-* and *multi-criteria optimizations* in the literature. The problem is called *non-linear optimization problem* if the objective function or constraints are non-linear. The solution of a problem can be a local or global optimum, which means for a global optimum that there is no other smaller/larger function value for all feasible points (global optimization). A local optimum is accordingly only the smallest/largest value for a specific neighborhood (local optimization). The reader may also refer to the available literature on optimization such as [Nocedal2006].

2.2.2 Different Optimization Algorithms

Optimization problems can be linear or non-linear problems and solved by a large variety of different optimization algorithms [Pham2000; Weise2009a; Blum2011; Nguyen2012]. Scientific workflows can in general belong to both problem types. This is due to their nature of defining a mixture of different algorithms, which can range from a linear function to a long running BLAST search [Altschul1990] or a complex structure prediction of protein folding [Floudas2007]. However, many used algorithms in the life sciences are non-linear problems [Bader2004], as their objective functions are not continuous and not differentiable with respect to the decision variables. Optimizing these algorithms means dealing with a rugged fitness landscape, maybe not including many gradient information. Therefore, only non-linear optimization problems will be part of the following investigation. Non-linear optimization problems can be solved by various optimization algorithms, which sample the search space to find a *good* (near optimal) solution in a reasonable amount of compute time. For single-objective optimization deterministic, stochastic or meta-heuristic methods can be used [Colaco2009]. Minimizing or maximizing a single-objective results in a one-dimensional vector of values. These values are also called *fitness values* and function as the measure of the capacity of each solution. Deterministic algorithms, such as gradient descent [Debye1909; Holtzer1954] or branch and bound [Land1960], are iterative methods hopefully converging closer to the optimum in each iteration by a specific step size. Simulated Annealing (SA) [Kirkpatric1983], which is a stochastic method but sometimes classified as heuristic method, aims at finding the global optimum by simulating the cooling process of materials.

Heuristic search methods [Pham2000; Siarry2008; Gendreau2010] are widely used in bioinformatics [Vesterstrm2005] such as Evolutionary Algorithms (EAs), in particular Genetic Algorithms (GAs) [Goldberg1988; Holland1992b], Particle Swarm Optimizations

(PSOs) [Kennedy1995; Engelbrech2005] or Ant Colony Algorithms (ACOs) for continuous problems [Dorigo1999]. Heuristic search methods aim at seeking near optimal solutions in a reasonable amount of time, however it is not claimed that they obtain the optimum with certainty. EAs simulate the process of biological evolution, by using selection, crossover and mutation to reproduce individuals and select the fittest individual. Individuals have chromosomes, which in turn have genes, which encode a particular solution. PSOs move around candidate solutions in an n-dimensional search space. A candidate solution is called particle, which is being moved related to its own best-found fitness and neighborhood's best previous position. ACOs were originally designed for discrete (combinatorial) optimization problems. Adapted to continuous problems they produce a population of ants representing several possible solutions, which can then be changed according to the pheromone density. The pheromone density is adapted by each ant within each step according to the efficiency of the solution. A detailed overview including references and examples for each of the presented algorithms is given in [Colaco2009].

Often it is required to consider several objectives in an optimization process. For example, a trade-off between sensitivity and specificity or between result quality and runtime. These optimization procedures manage two or more objectives and are called multi-objective or multi-criteria optimization. For multi-criteria optimization [Miettinen1999], the minimized or maximized optimal result is represented by a multi-dimensional vector of values. This vector of solutions is called the *Pareto optimal set* (PS). The image of this Pareto set in the objective function space is called *Pareto front* (PF). The PS comprises many solutions, as there exist different trade-off solutions, each describing a compromise between the different objective functions. Therefore, none of the solutions can be rated as the best with respect to all objective functions. Accordingly, a decision maker is required to select the Pareto optimal (ideal) solution or even a small set of optimal solutions because one single optimal solution does not exist for an optimization algorithm. A decision maker can be used during the optimization process or afterwards, although this is more critical during the optimization process [Branke2004].

Multi-Objective Optimization (MO) problems are commonly solved by using a Multi-Objective Evolutionary Algorithm (MOEA) [Zitzler2000; Zhou2011; Deb2001]. This is based on their population-based nature, which can be seamlessly adapted to MO problems. A challenge for each MO algorithm is the rating of the different solutions. Within multi-objective optimization, each solution has several fitness measures, one for each objective function so instead, each solution has to be rated regarding its *non-dominance* and closeness to the Pareto front. Additionally, methods to include diversity are required, which rates a solution as more suitable if it has no close neighbors. These aspects and some more, such as the classification of solutions, the selection, cross over and mutation mechanisms as well as elitism methods are implemented in many different fashions.

Approaches for PSO [ReyesSierr2006], SA [Suman2005], and other meta-heuristic inspired multi-objective algorithms have also been developed [Zhou2011], but they cause several more challenges than MOEA. On the one hand, it is more difficult to identify global and local best particles, which lead a population, and on the other hand, it is more difficult to maintain good particles and current non-dominated solutions.

There are some approaches to use multi-objective optimization methods for life science design problems [Handl2007]. Solving real-world problems such as molecular docking [Boisson2008], structure-activity relationship [Namasivaya2012], phylogenetic tree inference [Poladian2006] or drug discovery [Nicolaou2013] still poses great challenges for scientists due to the complex adaption of the problem transferal to an appropriate problem model or further to an appropriate approximation model. Therefore, multi-object optimization is not generally used in the life sciences. Similarly, multi-objective optimization has not yet been widely used for *parameter optimization* in the context of life sciences [Sun2012] in contrast to the engineering domain. However, it would be helpful to assess trade-offs, for instance between sensitivity and specificity or between result quality and runtime.

Within single and multi-objective optimization, hybrid approaches [Ishibuchi1996; Moscato1999; Blum2011] are becoming more popular, utilizing the different characteristics of deterministic and stochastic or evolutionary optimization algorithms in a joint usage. Primarily, two paradigms are typically applied with single-objective optimization, whereas a precise definition is still lacking [Blum2011]. One approach is a so-called integrative method, where one master optimization algorithm uses another embedded algorithm in order to substitute a specific function, such as the selection mechanism. Another type of method are so-called collaborative approaches, where two or more algorithms run in parallel and exchange information. Similar approaches exist for multi-object hybrid optimization methods.

2.2.3 Design Optimization Frameworks

In the engineering domain, many different optimization frameworks [Rao2009] have been established to optimize design models. These frameworks offer generic approaches to allow users defining own specific objective functions, constraints and the optimization algorithm by implementing a range of various common single- or multi-objective algorithms. Some of these frameworks are open-source available but most frameworks are proprietary software. Some of the open-source solutions also embed MATLAB [TheMath1994] in the back-end execution, which is a commercial software. Some other requirements for future multidisciplinary design frameworks were recently identified in [Parejo2012] such as parallel processing, user guides, project templates and graphical user interface integration, which are all not yet sufficiently supported [Parejo2012]. Other identified requirements aiming at monitoring, editing, and sharing of optimization problems have recently been included into engineering optimization frameworks as new features [Heath2012]. One of the most popular multi-objective generic frameworks, which provides a graphical user interface is the MOEA-framework [Hadka2012]. It offers a small range of multi-objective algorithms, but is extensible by several means. To implement new problems or algorithms, users are required to provide methods in Java or C++.

2.3 State-of-the-Art Scientific Workflow Optimization

The term *'scientific workflow optimization'* is used in several different contexts in the literature. Before discussing the approach of this investigation, this section reviews related work and gives a short overview of existing technologies.

2.3.1 Runtime Performance Optimization

One of the common optimization targets is to improve the scientific workflow runtime performance [Wieczorek2009]. As the typical scientific workflow is executed in e-Science infrastructures, several different approaches exist to intelligently schedule workflows or tasks on the Grid and Cloud [Yu2008]. These solutions are usually implemented for a specific SWMS and aim at the acceleration of workflows/applications through intelligent scheduling and brokering methods.

Scientific workflow runtime optimization is very heterogeneous as different criteria of the workflow can be taken into account [Wieczorek2009]. Typical criteria are the workflow structure, data processing or the component (task) model. Whereas workflow structure optimization aims at clustering or dividing jobs, data processing tries to optimize the execution of jobs regarding their data usage, and component model optimization schedules the single jobs regarding their task type.

Some of those solutions do not only support the optimization of for instance execution time and cost, but also other quality of service (QoS) parameters, such as reliability or security. The optimization extension for the WINGS/Pegasus SWMS developed by Kumar et al. [Kumar2010] focuses on the optimization of the runtime performance by modifying application parameters, which are likely to influence the runtime. The integrated framework takes quality of service requirements into account in order to adjust data dependent parameters so that, in a second step, the distributed data processing of applications can be improved. For example, some image processing applications have a parameter to adjust the *chunksize* of a data element. This chunksize can influence the runtime of a component and therefore needs to be optimized with respect to the quality of service parameter for picture resolution.

Other proposed mechanisms apply heuristic-based optimization algorithms to manage the scheduling of large-scale scientific workflows. These methods focus on user specified quality of service constraints for certain applications. A threshold can be set and the method then tries to find a solution so that all constrains are met and optimized. Available solutions usually target only fixed bi-criteria optimization and incorporate Ant Colony Optimization [Chen2009], Multi-Objective Evolutionary Algorithms [Yu2007] or Genetic Algorithms [Prodan2005] for scheduling or recursive loop handling.

Active Harmony provides a performance-tuning tool that can be used without in-depth domain knowledge [Chung2006] for changes. Users can change parameters of scientific applications or libraries to improve the performance on-line (during runtime) and off-line, respective to the parameters. The system requires developers to include tuning into the applications to specify the tunable parameters. Each of these methods is specified as a variable of an objective function that is optimized by a simplex method. The objective function can be defined by the user for example the number of processors used per node.

2.3.2 Output Performance Optimization

During the time of writing, no other comprehensive approach for general workflow optimization was freely available, with exception of Nimrod/OK [Abramson2010], which is a more specialized workflow parameter optimization tool from the engineering domain. Nimrod/OK extends the SWMS Kepler [Ludascher2006] with so-called *actors* to apply optimization in a scientific workflow. The optimization process itself is seen as a workflow that is composed of several steps. To run an optimization in Nimrod/OK, the user has to define a workflow consisting of a minimum of four tasks. These tasks comprise the set-up of a search space definer, points generator, the optimization process itself, and an optimization reporter. The computational model, which represents the objective function, is defined as a plain function expression or a MatlabNimrod actor execution [Enticott2010]. If a user wants to optimize a scientific workflow, it would require enveloping the workflow for it to be optimized into a second, optimization specific workflow. This mechanism lets the user design complex optimization workflows, which can combine various optimization methods within one workflow. In the available reports a simplex optimization, Hooke and Jeeves optimization and Genetic Algorithm [Lim2012] were presented as available optimization methods.

Another approach from the engineering domain for parameter optimization is using a workflow that itself implements the optimization process developed by Crick et al. [Crick2009]. Although Crick uses the Taverna management system [Missier2010] to design a parameter optimization workflow, this is a static rather than a general approach. The publication describes a manually created workflow, which comprises an optimization method for parameter optimization. Each cycle of optimization has to be defined manually in the outer optimization workflow.

2.3.3 Other Concepts of Workflow Modification

Sonntag and Karastoyanova [Sonntag2012] developed a SWMS which allows users to interact with the workflow during execution. It uses the Business Process Execution Language (BPEL) to define workflows and offers a graphical user interface for interaction. Especially if one task has failed, this process allows users to go back one step and modify the specific task. The main objective was to substitute the common procedure of trial and error for scientists so that they do not have to re-execute the entire workflow but only need to re-execute single tasks [Sonntag2010].

Another recent publication discusses the recommendation of scientific workflow compositions that assist researchers while building new scientific workflows [Koop2008]. In these approaches, different methods aim at suggesting several further steps to the user during the workflow design. More precisely, they show possible tasks or sub-workflows, which can be used to continue or complete the workflow. These suggestions are based on provenance analyses. The VisComplete system applies a graph-based approach, which analyzes the pipeline of direct acyclic graphs [Koop2008]. They extract all possible paths and additionally mine the input and output degrees for each vertex. The suggestions are provided to the researchers in a user-friendly way. Minor [Minor2011], Leake [Leake2008] and Chinthaka [Chinthaka2009] incorporate case-based reasoning for the analysis and the suggestion process of the components of a scientific workflow. Chinthaka manages different cases by requesting the user to add keywords to inputs and outputs as well as to the tasks. Leake et al. present a mechanism to mine workflow provenance and build cases from the gained meta-data. Minor and Görg [Minor2011] provide an approach that analyses workflow descriptions. These descriptions are available in an Extensible Markup Language (XML)-based data format and can thus be easily compared. They analyze a pair of diverse workflow or sub-workflow versions in order to identify the delta of difference to derive a case from this.

2.4 A Concept for Scientific Workflow Optimization

The last sections introduced the reader to the context and related work of scientific workflows and optimization. With this in mind, this section shortly revisits the motivation of this thesis and elaborates on a possible approach to investigate the capability of workflow optimization for the life sciences. This approach will function as basis concept for these thesis' developments presented in the next chapters (Chapter 3, Chapter 4, and Chapter 5).

Due to the rising complexity of scientific workflows [Barga2007], the common methods used by scientists such as trial and error or parameter sweeps are no longer feasible. To make the reuse of scientific workflows more worthwhile to scientists, an automated and intelligent method that can be applied with less effort is required. The target group for this project is made up of domain scientists and bioinformaticians. Researchers from the first group are typically inexperienced with computer usage. Thus, the concept for workflow optimization will not only determine technical preferences but also establish aspects to make optimization fast, robust and handy to be adopted by life scientists. An important point is that scientists typically stick to their accustomed technologies in order to avoid dealing with yet another technique and another user interface. Violating this may indeed be one reason for life scientists not applying existing optimization frameworks from the engineering domain. In e-Science infrastructures, SWMSs turned out to be the standard solution for life scientists to manage scientific workflows visually. Thus, just as Abramson [Abramson2010] and Kumar [Kumar2010] implemented, the approach for scientific workflow optimization should ideally be incorporated to extensible SWMSs. SWMSs also provide mechanisms for provenance capturing and sharing purposes, which may also be required to reproduce and understand optimized scientific workflows.

Many design frameworks provide programming interfaces or configuration files to set up an optimization process. Additionally the workflow complexity expands during the optimization process, as it is the case in Nimrod/OK [Abramson2010]. Contrary to these aspects, the complexity of workflow optimization is aimed to remain hidden from the user within this thesis approach. The main focus has to be the optimization process rather than scripting configuration files or designing another workflow that performs the optimization.

Summarizing these requirements as a starting point for the concept approach implies the following: Scientific workflow optimization should be investigated in a workflow management system, offering an easy-to-use graphical interface. The optimization process should not alter the workflow complexity.

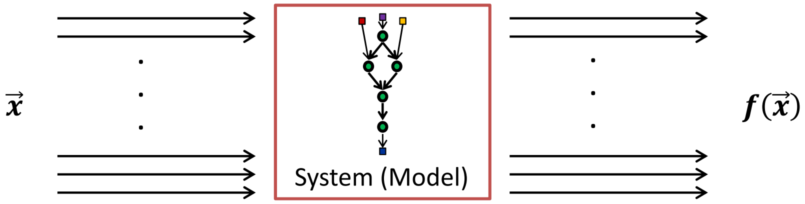


Figure 2.2: A workflow in the context of optimization. One workflow execution corresponds to one evaluation of a model.

Conventionally the model (fitness function) is represented by a real function, an algorithm, or a simulation. In these cases, the optimization process has to deal with the parameters of a single function, algorithm, or simulation. Considering the optimization of scientific workflows (cf. Figure 2.2), it is not assessable which algorithms or simulations will be used in a workflow. The used algorithms and simulations in scientific workflows are typically non-linear problems [Bader2004]. They may have arbitrary functions and the optimization algorithm has to deal with different parameters. Thus, a workflow optimizations' fitness landscape may be rugged and not assessable. Figure 2.2 illustrates the experimental set-up of workflow optimization that this thesis is based on.

In order to to deal with these issues properly, robust and versatile optimization techniques are required, such as meta-heuristic search methods. They provide optimal (sometimes near optimal) solutions in a reasonable time for real-world non-linear problems [Alba2013]. Compared to parameter sweep approaches, these meta-heuristic algorithms usually need to explore a much smaller part of the parameter search space in order to find good solutions. Additionally, parallelism mechanisms can be applied to many heuristic search methods in a natural way to reduce the search time [Alba2013], which is very important with respect to workflow execution times. Summarized, workflow optimization should be handled by heuristic search methods within this thesis approach.

The described model assembly of workflow optimization (cf. Figure 2.2) points out that the optimization algorithm has to deal with the parameters of several different algorithms or functions. Moreover, as the whole workflow is assimilated into the optimization, one may also think of other workflow description levels that could be targeted by the optimization algorithm. Examples are the used functions and simulation programs or the implemented topology. Thus, the optimization of scientific workflows cannot only be defined by an optimization problem but also by the level of workflow optimization. The investigation of different workflow levels should be handled by workflow optimization and will be defined more precisely in Chapter 4.

Engineering design frameworks offer a large variety of optimization algorithms. This is based on the no-free-lunch theorem [Wolpert1997], which says that all algorithms yield similarly good results for all optimization problems in the average. Thus, an interested researcher may want to test different search methods such as Genetic Algorithms, Particle Swarm Optimizations or Ant Colony Optimizations as well as single- or multi-criteria optimization. Moreover, users may want to apply improved and novel techniques. Developers who are willing to provide these techniques are typically not familiar with workflow systems or usage of distributed computing resources. They therefore do not hold an interest in providing novel methods promptly. To prevent them from developing a bottom-up solution every time, they require a platform offering an abstract and general mechanisms to easily provide workflow optimization methods for different algorithms and levels. Summarizing this, workflow optimization should be possible in a generic way and developers should ideally provide these options. However, developers require crucial support for provisioning.

Scientific workflows in the life science domain can become very complex and computationally intensive. The execution of a workflow might last anything from a few minutes up to several hours. In order to perform an optimization according to the problem set-up in Figure 2.2, many different workflow instances might be executed which sum up the required time for optimization. The required time for the optimization process can certainly be reduced by utilizing only a sample of the complete data set. This reduction should be applied, especially when using an example data set for testing purposes. However, to keep the time required for optimization of any data set as short as possible, different acceleration strategies should be devised.

One aspect may be accelerating the execution time of single components. This can be achieved not only by moving data parallel, but also by moving time consuming and parallel tasks to high performance computing recourses. Consequently, if a parallel version of a component is available it should preferably be used. Computing resources can be accessed via middleware systems in e-Science infrastructures. Therefore, middleware support requires to be established in SWMSs. Certainly, security standards for secure access and scalability capabilities must be incorporated for a reasonable support. On the other hand, the optimization process itself should be executed in parallel. The execution of several workflows in parallel still represents a challenge [Gorlach2011] but should be enabled in this thesis concept. Again, High-Performance Computing (HPC) resources and middleware, which are available in e-Science infrastructures can be employed. This is also enabled by Nimrod/OK [Abramson2010], wherein parallel execution was identified as a crucial missing feature within most engineering design frameworks [Parejo2012].

Summarizing these insights, the parallel execution of components and workflows is needed and middlewares state as applicable solution to access distributed data and computing resources.

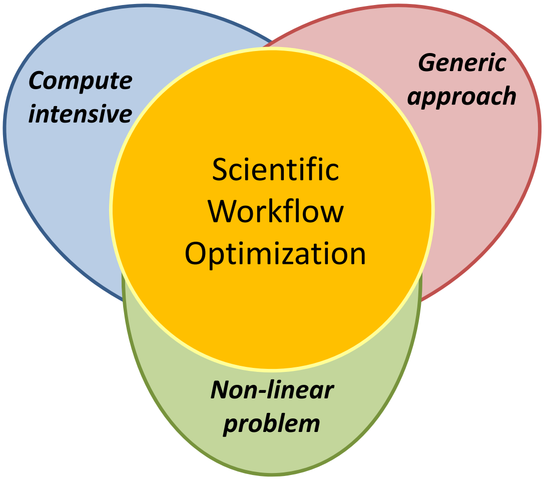


Figure 2.3: The three different requirements to investigate workflow optimization.

Summarizing the identified requirements of scientific workflow optimization, the approach of this thesis to investigate scientific workflow optimization is divided into three parts as illustrated in Figure 2.3:

1. Solve workflow optimization in a robust way → Non-linear optimization problem must be tackled by a reasonable optimization algorithm.

2. Provide abstraction and integration mechanism for different optimization algorithms and levels→ Multipurpose, extendable and generic approach have to be developed.

3. Need to fulfill acceleration and computational demands for optimization → Complex and compute-intensive workflows require parallel execution.

The last requirement - parallel execution of compute-intensive workflows - may be implemented by leveraging computing resources provided by e-Science infrastructures. These offer access to various distributed computing resources, i.e., High-Performance or High-Throughput Computing resources for instance supercomputers or clusters. The execution of a workflow task and also the execution of an entire workflow must be feasible within a workflow management system. An approach to this is presented in Chapter 3.

The second requirement - a generic approach - could be put into practice by following the principals of engineering solutions: a framework-based approach. A framework can warrant the advantages of being generic and quickly extendable along with providing a stock of basic features by an abstraction layer to minimize the programming effort for further extensions. Chapter 4 discusses an example of its implementation.

The first requirement - non-linear optimization problem - could be tackled by utilizing a meta-heuristic based optimization method to sample the search space intelligently. Heuristic methods are popular for being time efficient and thus providing good (optimal) solutions. This approach is investigated in Chapter 5.

In the following chapters, concepts are formulated and developed regarding the evolved requirements and goals. More precisely, to test the feasibility of a general approach, an SWMS will be extended by appropriate methods to finally experimentally test scientific workflow optimization.

Chapter 3

Enabling Parallel Execution in Scientific Workflow Management Systems

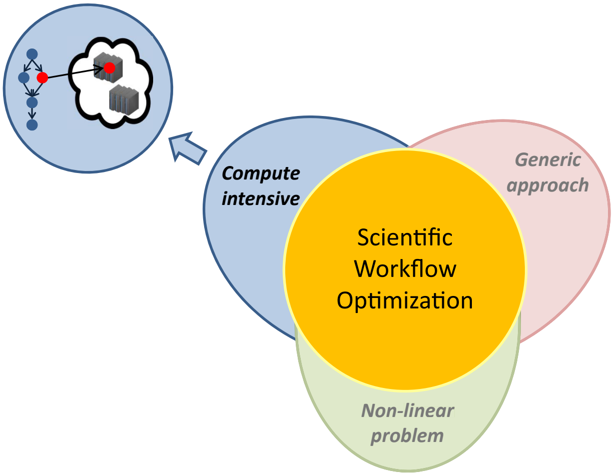


Figure 3.1: The three tier concept for workflow optimization. First approach: acceleration of compute-intensive applications.

Within the last chapter, scientific workflow management systems were selected to assess the implementation of scientific workflow optimization. Due to their versatile implementations and characteristics, the first section of this chapter focuses on the investigation of workflow systems to determine one workflow system which shall be used as reference implementation within this thesis. In the following sections this workflow management system will be adapted to meet the first requirement of the proposed approach, namely the acceleration of compute-intensive applications to perform optimization in a reasonable amount of time (cf. Figure 3.1).

In the last decades, compute-intensive algorithms and simulation software took considerable advantage of High-Performance Computing (HPC) resources that accelerate execution by parallel processing methods. E-Science infrastructures provide large-scale and distributed computing resources to enable data- and compute-intensive experiments [Hey2002]. Some were successfully establishment such as the Distributed European Infrastructure for Supercomputing Applications (DEISA) [DEISA2013] or the eXtreme Science and Engineering Discovery Environment (XSEDE) [XSEDE2013]. Access to these computing infrastructures is mainly provided via middleware systems, such as UNICORE [Streit2010] (UNiform Interface to COmputing REsources), ARC [Krabbenhof2008] (Advanced Resource Connector), or Globus Toolkit [Foster2005]. Following, using such technologies combined with SWMSs is a possible approach to accelerate compute-intensive workflow tasks, which are employed as parallel executions on distributed systems. This chapter describes the investigation and development of the embedding of a middleware into a workflow management system. This extension aims to allow the submission of workflow tasks to distributed computing resources for parallel execution.

3.1 Investigation of Scientific Workflow Management Systems in e-Science

Scientific workflows have received considerable interest from scientists within the last years. They support the aggregation of complex and distributed e-Science applications to perform *in silico* experimentation. SWMSs, introduced by Vouk and Singh in 1996 [Vouk1997], provide a graphical user interface as well as a workflow engine to model, develop, execute and monitor scientific workflows. During the last years around 50 SWMS appeared [Goble2009], each of which serves different needs such as parallel execution, semantic support or special data and provenance aspects; to name only a few of the most popular ones: Taverna [Missier2010; UoM2009], Kepler [Ludascher2006; NSFKepler2005], Triana [Allen2003; Cardiff2012], Pegasus [Deelman2005; USC2010], KNIME [Berthold2008; KNIME2013], Galaxy [Goecks2010; Galaxy2013] and Pipeline Pilot [Yang2010; Accelrys2007]. Many of these systems serve the life science domain [Achilleos2012] and provide a uniform access to computational as well as data tools. They are available as open source or commercial software. Each has its own workflow language to describe workflow compositions and models of computation. The languages come along with a workflow engine, which interprets scientific workflow descriptions and invokes the components. Both will not be pursued in the further course of this thesis study. The execution of workflow tasks is typically performed on a scientist's laptop or desktop computer but can likewise be executed by a Web service on a compute cluster or distributed computing environment.

In order to show that a general approach for workflow optimization is feasible, this thesis aims to develop a reference implementation within one SWMS. Due to the wide range of available SWMSs, this section provides a brief overview of the most popular ones and closes with a detailed evaluation of the most appropriate SWMS for scientific workflow optimization within the life science domain. A more detailed outline as well as comparisons of SWMSs can be found in [McPhillips2009; Ludascher2009a; Deelman2009; Curcin2008; Yu2005; Yeltayeva2012].

Kepler

The Kepler workflow management system [Ludascher2006] is based on the Ptolemy 2 framework, an actor oriented design. Actors are small local services represented by scripts, programs or other software that can be executed on a client machine. The Kepler system comes with a wide range of around 350 actors for various tasks. As Ptolemy is used in the physics domain, Kepler was adopted by the physics community. Kepler workflows are executed via so-called directors, which manage the execution mode of an actor, such as synchronous dataflow or process networks. Kepler is extendable by the means of actors and directors. Several extensions have been developed especially for the physics domain, but also for bioinformatics and chemistry [Wang2010; Stropp2012].

Triana

Triana was originally developed within a gravitational wave detection project, GEO 600 [Balasubram2005]. In 2002, it became part of the GridOneD project [Taylor2002], aiming at supporting distributed application executions on Grid infrastructures. Furthermore, it was extended with peer-to-peer communication and many generic Grid adapters [Allen2003; Taylor2003] to tightly couple Grid services and Web services. In general, it contains several local processing units, representing scripts and programs. Processing units target one-dimensional data transformation as well as visualizations and over 500 components for signal, image and audio processing [Taylor2006b].

Wings/Pegasus

Wings [Gil2011] is a semantically driven workflow system focused on the creation of workflow templates. It is used in conjunction with Pegasus, which maps abstract components to discrete applications, data objects or execution resources. The Pegasus system [Deelman2005] offers a graphical user interface based on the Eclipse Rich Client Platform. The main focus of Pegasus is to support distributed computing resources, especially the middleware Condor, Globus, or Amazon EC2 [Juve2012b].

Taverna

The Taverna workflow management system [Missier2010] is mainly focused on life science applications to satisfy the needs for scientists. It is equipped with a wide range of available services on startup (e.g. database access or NCBI services [Geer2010; NCBI2009]). These services are basically Web services, partly Java scripts, local programs or Grid jobs. Taverna is part of the myGrid [UoM2008a] e-Science initiative, which provides other service directories, search tools and several meta-data repositories (e.g. BioCatalogue [Bhagat2010; UoM2008b]).

Pipeline Pilot

The Pipeline Pilot workflow application [Yang2010] is a commercial framework enabling the access, analysis and reporting of scientific data. It is focused on the abstract representation of distributed computing and data resources for various domains such as chemistry, chemoinformatics or bioinformatics. Also, it comes along with an extensive number of services for such domains.

KNIME

The KNIME *Konstanz Information Miner* [Berthold2008] is a free-to-use workbench, providing various functionalities for data analysis. It comes with many ready to use components, some of them aiming at data mining for drug discovery, bioinformatics or chemistry, especially developed within the last years [Jagla2011; Lindenbaum2011]. The KNIME products, which are not free of charge, let users employ additional services such as shared repositories, security mechanisms, remote server or cluster execution and scheduling. The source code is not freely available and only component categories and component functionalities can be extended by third parties.

Based on the theoretical concept for scientific workflow optimization proposed in Chapter 2, eight key requirements were distinguished to evaluate the proposed SWMS:

- Support of life science applications (Web and local services)

- Provision of an Graphical User Interface (GUI)
- Seamless integration of heterogeneous resources such as computing or data
- Support for workflow provenance and sharing
- Scalability and parallel job submission to ensure acceptable workflow execution time
- Easily extensible in many ways
- Support of security mechanisms
- Under active development

	Kepler	Triana	Wings/ Pegasus	Taverna	Pipeline Pilot	KNIME
Life Science application support	++	++	+	+++	++	+++
GUI	+++	+++	+++	+++	+++	+++
Heterogeneous resources	+++	++	++	+++	++	+
Provenance & Sharing	++	++	++	+++	++	+
Scalability	+++	++	+++	++	++	++
Extensible	+++	++	++	+++	-	+
Security	++	+	+	++	+	+
Active development	+++	++	+++	+++	+++	+++

Table 3.1: Evaluation of common scientific workflow management systems. Legend: '-' means not supported, '+++' means fully supported.

In general, all presented workflow management systems provide a graphical user interface to support especially less experienced computer users. Taverna and KNIME offer large-scale life science support, whereas Pegasus provides enhanced support for distributed computing resources. Kepler and Taverna supplementary offer many data resource services.

All workflow management systems provide the ability to store provenance and provide sharing mechanisms, albeit Taverna offers access to the workflow repository myExperiment [Goble2010] from within the workbench. Scalability can be achieved by almost all systems, whereby Kepler and Wings are more suitable for distributed computing. All but Pipeline Pilot are open source and can be extended in many ways. Taverna and Kepler support enhanced security mechanisms and all except Triana are under active development.

A detailed analysis of all requirements and evaluations is presented in Table 3.1. It highlights the properties of each SWMS regarding the requirements to fulfill the aim of this thesis. Based on the evaluations, Taverna emerged as the most appropriate SWMS for the life science domain to assess scientific workflow optimization and will be used as a reference implementation in this thesis.

3.2 Extension of a Workflow Management System for Highly Parallel Workflow Execution

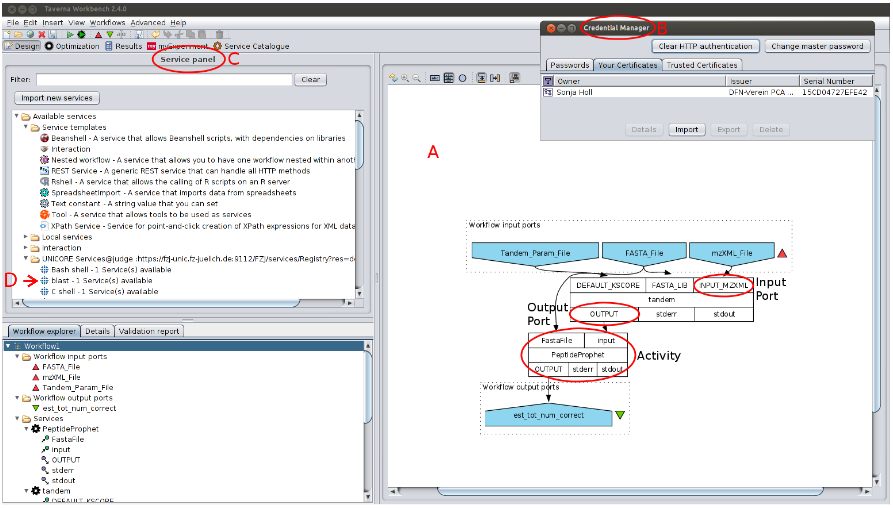
The SWMS Taverna has been widely used in the life science domain for many years and was taken as a reference implementation in this thesis. It enables users to create, develop and execute scientific workflows in e-Science infrastructures. Life science applications are partly compute-intensive and require acceleration to meet current and future requirements of data analysis [Zhao2008] and optimization as undertaken within this thesis. Taverna already offers several services to satisfy these needs [Tan2010; Maheshwari2009]. This section will focus on the integration of UNICORE 6 [Streit2010], a middleware providing a seamless and secure access to distributed compute resource, into the Taverna workflow system. Among other features, UNICORE supports a special environment for Message Passing Interface (MPI)-based jobs and is therefore very suitable for accessing High-Performance Computing (HPC) resources, which are represented in a highly abstracted manner. The approach designed and developed in this section is in detail described in [Holl2011] and differs in existing solutions: "The development allows researchers to use UNICORE in Taverna to access any infrastructure, regardless of which type of computing or data resource. Researchers are not required to provide any resource information, for example executables, arguments, or file paths. The presented development provides only services, which descriptions are taken directly from the corresponding resource. As a consequence, instantiated services are guaranteed to meet the requirements of the execution resource. Looping is not necessary for monitoring the job status. Finally, the presented solution includes a high-level security policy based on X.509 certificates." (Modified from [Holl2011].)

To provide the reader with an appropriate background on Taverna, the distributed computing infrastructures and especially on the middleware UNICORE, the first part of this section introduces the Taverna workflow management system and the UNICORE middleware. In the following, the approach developed within this thesis for distributed computing support is described. It comprises a plugin for Taverna that shall enable parallel submission of workflow tasks to the UNICORE 6 middleware and thereby provides access to distributed computing resources for domain scientists (e.g. biologists), who often are non-expert computer users. This section was already published and parts were adopted from [Holl2011].

3.2.1 The Taverna Workflow Management System

Taverna [Missier2010; Wolstencro2013] is available open-source, written in Java and under active development by the School of Computer Science, University of Manchester. It is equipped with around 3500 services [UoM2009] for the life sciences, which are all available via the so-called *Taverna Workbench* that provides a graphical user interface shown in Figure 3.2. In the workbench researchers can use service providers (vie the *Service Panel*, cf. Figure 3.2(C)) to design a workflow (in the *Workflow Diagram*, cf. Figure 3.2(A)), configure the workflow, provide input values, monitor the workflow execution, obtain results and access myExperiment. The execution is managed by the *Taverna Workflow Engine*, which is either integrated in the client or available as stand-alone Taverna Server. A *Credential Manager* (cf. Figure 3.2(B)) can be used to set passwords, certificates and trusted certificates to fit different security settings.

The components of a Taverna workflow are called *activities* (cf. Figure 3.2) and are connected as a data-flow. Each activity provides so-called input and output *ports*, which can be connected to define the workflow composition. The execution of a service is initiated by its inputs; once all required inputs are available, the execution is started. If the cardinality of the inputs is higher than defined for the port, Taverna creates a separate invocation of the activity for each input, and can run them in parallel, cf. Figure 3.3. If several input cardinalities are higher, the cross or dot product of all input values is executed in parallel. The type of algebraic operation as well as the maximum number of parallel executed services can be manually defined by the user for each service individually.



- Available services
  - Service templates
    - Beanshell - A service that allows Beanshell scripts, with dependencies on libraries
    - Interaction
    - Nested workflow - A service that allows you to have one workflow nested within anotl
    - REST Service - A generic REST service that can handle all HTTP methods
    - Rshell - A service that allows the calling of R scripts on an R server
    - SpreadsheetImport - A service that imports data from spreadsheets
    - Text constant - A string value that you can set
    - Tool - A service that allows tools to be used as services
    - XPath Service - Service for point-and-click creation of XPath expressions for XML data
  - Local services
  - Interaction
  - UNICORE Services@judge :https://fzj-unic.fz-juelich.de:9112/FZJ/services/Registry?res=d
    - Bash shell - 1 Service(s) available
    - blast - 1 Service(s) available
    - C shell - 1 Service(s) available

- Workflow1
  - Workflow input ports
    - FASTA_File
    - mzXML_File
    - Tandem_Param_File
  - Workflow output ports
    - est_tot_num_correct
  - Services
    - PeptideProphet
      - FastaFile
      - input
      - OUTPUT
      - stderr
      - stdout
    - tandem
      - DEFAULT_KSCORE

Owner	Issuer	Serial Number
Sonja Holl	DFN-Verein PCA ...	15CD04727EFE42

Figure 3.2: The Taverna Workbench showing (A) the Workflow Diagram, (B) the Credential Manager, and (C) the Service Panel. The Service Panel shows available UNICORE services (D).

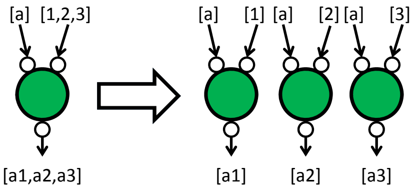


Figure 3.3: Taverna parallel execution: The left hand side shows an activity, whose input port can handle only one data item (depth 0) but receives an array of items (depth 1). Concealed from the user, Taverna creates a set of parallel activities (right hand side) whereof each one handles one of the items in the array. After execution, the outputs are in turn concatenated to an array. The dot product for item concatenation was demonstrated in the example.

The Taverna system also interfaces with many projects. A major extension offers access to the myExperiment workflow platform [Goble2010], which is a well-known e-Science workflow repository allowing researchers to share, store and reuse scientific workflows. Access to BioCatalogue [Bhagat2010] is also provided by Taverna, which is a supervised catalog for Web services especially for the life science domain. Taverna also allows integrating BioMoby [Kawas2006] services, which offer a large collection of Web services for the bioinformatics domain. Several approaches have been developed to make Galaxy [Goecks2010], a Web-based portal for genomic research, and Taverna interoperable [Abouelhoda2010]. Recently it is possible to i) execute Taverna workflows within Galaxy and ii) execute a combination of Galaxy and Taverna workflows within an e-lab portal solution in the Cloud [Abouelhoda2012].

3.2.2 UNICORE Middleware

UNICORE 6 (UNiform Interface to COmputing REsources) [Streit2010] is a platform-independent, open-source Grid middleware that offers a seamless, secure and intuitive access to e-Science infrastructures including distributed computing, data resources and other scientific instruments. Examples of production Grids are the German D-Grid Infrastructure[D-Grid2013], the European DEISA/PRACE infrastructure [DEISA2013], the Belarusian-Russian SKIF-Grid infrastructure [Lukoshko2010] or the National Science Foundation cyberinfrastructure XSEDE [XSEDE2013]. UNICORE conforms to many common open software and Web standards [Baker2005] in order to allow a mature and stable access to computing and data resources. Security standards are strictly applied to enable a secure access. The UNICORE middleware provides computing and data abstraction to a wide range of computational resources such as high performance computing (HPC) supercomputers or high throughput computing (HTC) oriented geographically distributed resources. The resource management abstraction provides a simple but efficient interface hiding the low-level details from the end-users. One *UNICORE site* represents one server and one target system for job and data management. Important UNICORE components for this thesis are shortly described in Table 3.2.

3.2.3 Architecture of the Grid Plugin

The Taverna management system features a plugin-based mechanism and several extensible interfaces. All plugins have to follow strict Taverna guidelines regarding the Service Provider Interface (SPI) to be accessible during startup. The most frequently used extension point is the *activity service point*. New services can be added to the activity service provider to then be utilized in a workflow during design and execution.

Full Name	Abbreviation	Function
Service Registry	Registry	Global entry point for requests available at a specific service registry address; registers all UNICORE services and sites.
Service Orchestrator	SO	Handling of job executions and monitoring. Workload balancing is implemented by different brokering strategies.
Grid Resource Information Service	GRIS	Collection and publication of individual meta-information for the different sites (resources and applications).
eXtended Network Job Supervisor	XNJS	The XNJS is the main execution component, performs file transfers, and validates incoming requests based on compute and data capabilities available on back end resources.
Incarnation Data Base	IDB	Mapping of abstract job descriptions to job executions. Stores application meta-data, such as input and output parameter, type information, default values, dependencies or valid value ranges [Demuth2010].

Table 3.2: Description of important UNICORE services, which have been used to implement Grid access though Taverna.

Corresponding to the Taverna guidelines, the plugin consists of one large component including both the *service activity* and *service activity user interface*, as shown in the architecture Figure 3.4. A third, component was developed that realizes UNICORE functionalities, which are called up by the service activity and user interface in case of UNICORE-specific queries, submissions or file transfers. Additionally, this component contains UNICORE libraries, necessary for communication between the services in Taverna and the UNICORE middleware. Together, these three implemented units build the UNICORE-Taverna plugin.

3.2.4 Development of the Grid Plugin

In order to access e-Science infrastructures via UNICORE, the user and server certificates are required for the authentication process due to UNICORE's security standards [Benedy-

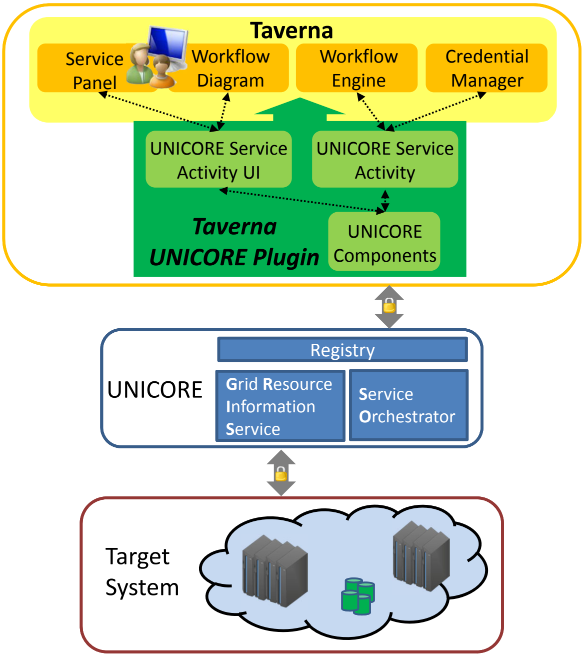


Figure 3.4: System architecture of the UNICORE-Taverna plugin. (Source: [Holl2011])

czak2011]. In order to let users deal with the common security mechanisms available in Taverna, the Taverna Credential Manager (cf. Figure 3.2) was used to access the X.509 user certificate and trusted server certificate authorities (CA). As entry point to Grid applications, users can insert a Grid registry address (cf. Table 3.2). Via the registry, the GRIS (cf. Table 3.2) is queried by the plugin and returns a complete list of available applications from registered UNICORE sites that includes meta-data such as name, version, and detailed information on the inputs and outputs (data types, default values, valid values, etc.). The meta-data specification was only recently made available and is described in [Demuth2010]. The application list is displayed in the *Service Panel* (cf. Figure 3.2) as a list of new service activities to be used within a Taverna workflow. The application meta-data are used to *automatically* create the activity instance and to compose inputs and outputs in the *Workflow Diagram*. They also allow distinguishing between different implementations of the same application on the Grid. For example, the UNICORE activity *tandem* comprised in the workflow shown in Figure 3.2 was built of the available meta-data (minimal example) in Listing 3.1. Using this mechanism, developers and administrators can provide new or extended applications in a fast and easy way to the users. By providing this meta-data description all users can immediately use the application without any further programming.

```
<ApplicationName>tandem</ApplicationName>;
<ApplicationVersion>0.1</ApplicationVersion>;
<Argument Type="filename" Default="param.xml">DEFAULT_KSCORE</Argument>;
<Argument Type="filename" Default="input.mzxml.zip">INPUT_MZXML</Argument>;
<Argument Type="filename" Default="fasta\_lib.fasta.zip" >FASTA_LIB</Argument>;
<Output Type="filename" Default="output.zip">OUTPUT_ZIP</Output>
```

Listing 3.1: Tandem XML representation. The plugin creates a GUI element and model for execution by interpreting the displayed meta-data. The *Argument* type can vary between *int, double, string, choice, filename*, and *fileset*. Other meta-data attributes are *IsMandatory, IsEnabled*, or *ValidValues*.

On execution of the UNICORE activity, local files specified as inputs are automatically uploaded to a storage on a UNICORE site. The plugin creates a standard-conform job document, which is described by the Job Submission Description Language (JSDL). It specifies the execution details of the application and is submitted via Web services to the SO (cf. Table 3.2). The SO manages the execution, file stage-in, workload balancing and monitoring. On the client side, the plugin monitors the job execution in the background hidden from the user by continuously polling the job status.

After successful execution, outputs are automatically downloaded as a string value, as a file or as logical file address. Accordingly, the string can be directly forwarded to the next activity, a file will be stored on the user's computer or the file remains on a UNICORE storage. Error handling is accomplished on the one hand by providing the stdout (standard output) and the stderr (standard error) and on the other hand by passing evaluated server side error messages to the user.

3.2.5 Enhanced Parallel Application Execution by Sweep Job Submission

Taverna offers an easy to use parallelization mechanism for activities, by referring to the depth of an input port and the incoming data object as described in Section 3.2.1 and Figure 3.3. The UNICORE-Taverna plugin introduced in the last sub-section manages this *parallel* execution of activities by submitting a single job instance for each input set. For example, if twenty activity instances are created (by using Taverna's parallel execution mechanism shown in Figure 3.3), twenty individual jobs are submitted to the computing infrastructure.

The mechanism works well, unless exceeding a certain number of parallel executions. Many parallel executions reduce the performance as a high amount of Web service calls do not only produce a high workload on the client and server, but also slow down the submission and monitoring process because of network traffic. UNICORE developers recently identified this problem and developed a UNICORE server extension to execute parameter sweeps within one job description according to the open standards based job description parameter sweep extension [Drescher2009]. The extension as well as the remainder of this section is published in [Memon2013] and [Holl2013d].

The previously developed UNICORE plugin was extended further within this thesis to adapt the new UNICORE server sweep mechanism. Therefore, a sweep generator was developed as shown in Figure 3.5. It collects all instantiated activities before the job submission takes place. The instances are aggregated by translating each original job description into one larger job description. In order to upload globally used files only once the sweep generator analyses all input port values. If values of the same port are identical, a global parameter value is set. If the values differ, an individual parameter value is set for each iteration within a so-called *sweep node*. This means, that each sweep node hosts one of the individual parameter values in the job description document.

When the job script is submitted to a UNICORE site, the newly extended sweep library reconstitutes the original activities and creates as many job descriptions as determined by the sweep nodes. These jobs are then submitted to a target system for execution, whereas each of these executions (child job) is independent but nevertheless monitored by the same master job. After successful execution of the child jobs, the master job changes its state and the Taverna-UNICORE plugin downloads the outputs as usually as: string, file or local file address. The sweep manager sorts the individual results back to the corresponding original activities and the workflow resumes.

3.3 Evaluation by Life Science Use Cases

To evaluate the performance of the UNICORE-Taverna plugin presented in Section 3.2, an examination with an example workflow taken from the proteomics domain was conducted. In the following, the use case is described in detail as well as a survey presenting local, parallel and sweep job distributed execution.

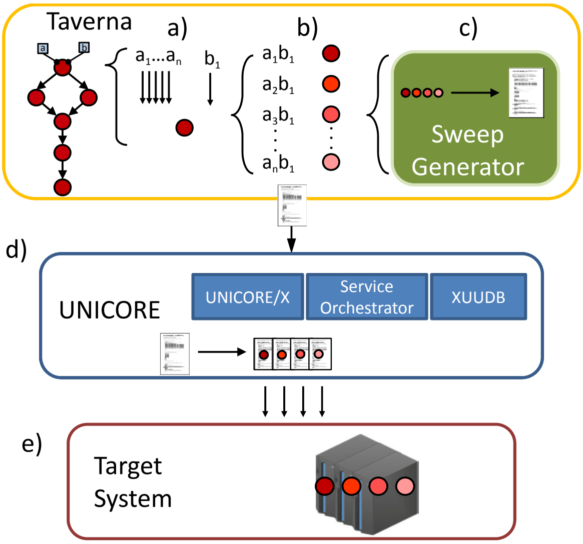
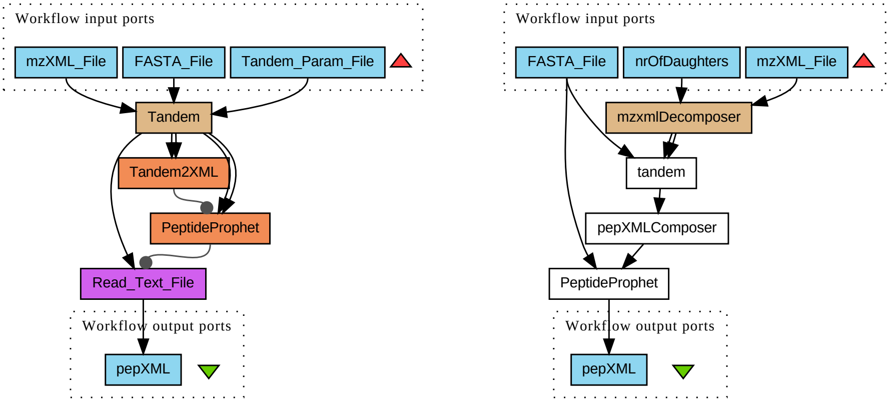


Figure 3.5: The figure shows the sweep job mechanism in Taverna. (a) One value set and one single input value arrive at the workflow input ports. (b) Corresponding to the parallel execution in Taverna, several parallel activities are instantiated. (c) The sweep generator collects all instances and creates one job sweep description for submission to UNICORE. (d) The UNICORE server splits up the job description script into the original number of executions. (e) The applications are executed on the target system.

In proteomics, fundamental questions are targeted by analyzing proteins experimentally to understand their structures and functions. Mass spectrometry is one of the used techniques to identify the characteristics of proteins. This technique measures individual masses of protein molecules in a sample to determine its composition and to draw conclusions about structures and functions. The workflow shown in Figure 3.6 (a) was developed within the proteomics domain to identify proteins. This ’bottom-up’ identification of proteins via the enzymatic digestion of the proteins into peptides is a typical task in mass spectrometry (MS)-based proteomics [Aebersold2003].

After the digestion of a protein, the peptides are separated by liquid chromatography coupled to mass spectrometry (cf. Figure 3.7 *MS1*). In liquid chromatography, the peptides are separated by their physicochemical properties, such as size, charge and hydrophobicity. In the mass spectrometer, the peptides are separated with very high resolving power based on their mass-to-charge ratios. Individual peptides are then selected and fragmented using



(a) Local X!Tandem workflow

(b) Grid X!Tandem workflow

Figure 3.6: The X!Tandem workflow (a) for local execution on a client machine. Workflow (b) executes the compute-intensive X!Tandem application on the Grid. The Tandem_Param_File is created by the mzxmlDecomposer

collisions with neutral gas or ion-electron reactions (cf. Figure 3.7 *MS2*). This complete procedure is called *'tandem mass spectrometry'* or *'MS/MS'* [Aebersold2003]. The output is a set of tens of thousands of peptide masses and associated fragmentation spectra (mass-to-charge ratio). These measured spectra can then be compared to previously calculated spectra from the genome by one or more algorithms – here X!Tandem [Craig2004] – to identify proteins. For the identification, X!Tandem compares measured spectra to calculated spectra for all peptides of similar mass that could possibly be generated by the enzyme used from the biological species. After this step, PeptideProphet [Keller2002] estimates the probability of each peptide-spectrum match assigned by X!Tandem by a mixture model of the X!Tandem score distribution, assuming there will be some correctly and some incorrectly identified spectra. This result can be used to measure the identification process. X!Tandem as well as PeptideProphet are available within the Trans-Proteomic Pipeline (TPP) [Keller2005].

The workflow shown in Figure 3.6 (a) accomplishes the execution of X!Tandem and PeptideProphet on a local client machine. X!Tandem uses the database as input (*FASTA_File*), which stores the calculated spectra for all peptides. Another input file contains all the measured spectra (*mzXML_File*). The workflow output port returns a summary file that contains among others the number of correctly identified peptides given by PeptideProphet,

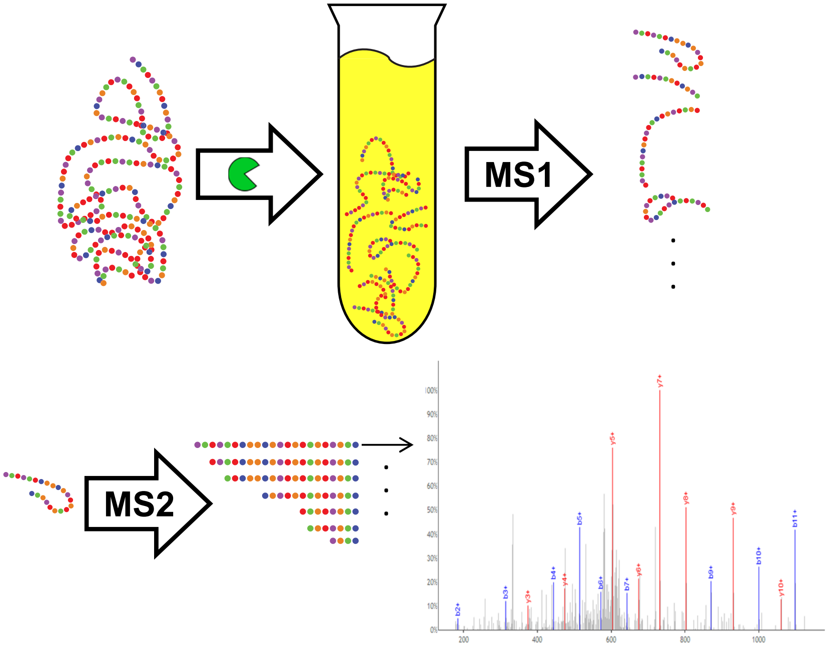


Figure 3.7: The concept of tandem mass spectrometry. First a protein is digested and split into peptides (upper left side). Afterwards, the peptides are separated by their physicochemical properties (MS1). These peptides are again split into fragments(MS2). The fragment spectra (lower right side) measure the mass-to-charge ratio.

which statistically evaluated the identified measured spectra. Figure 3.6 (b) shows the same scientific experiment, ported to an distributed computing environment. Therefore an mzXMLDecomposer and pepXMLComposer, developed by Mohammed et al. [Mohammed2012] were used to decompose the original input mzXML file into several small input files. X!Tandem can then be executed in parallel, where each invocation compares the measured spectra of one piece of the input file. The pepXMLComposer joins the results after the X!Tandem executions. The spectra and the used database, which stores the known peptides, can be of differential size between 20 MB - 200 MB for both. The local execution of the workflow was performed on an Intel i5-2520M processor, 2.50GHz, 4 CPUs, 8GB RAM. The compute cluster, on which the parallel execution was executed, has 206 compute nodes, each of which consists of 2 Intel Xeon 6-core processor with 2.66 GHz and 96GB main memory. For the execution, 4 CPUs per job were requested.

	Common parallel execution	Parallel sweep execution
Total file upload in MB	1312.0	151.2
Total Web service calls	4032	126
Average CPU load on the client in %	43.7	14.1
Total network packets	458,353	15,244
Totally transferred bytes	1,370,513,098	190,571,550

Table 3.3: Statistics of the submission via the conventional submission mechanism and the developed sweep generator.

In the presented example, the used sequence database consists of 37.4 MB sequences and the input data set consists of 113.8 MB spectra. To speed up the execution, the input file was split into 32 sub files (file size between 3.0 MB and 4.2 MB). The non-sweep classic solution submits one job to the computing infrastructure for each of those sub-files (in total 32). Each of these jobs consists of the sequence database and one sub file (41 MB upload per job). Each of these 32 jobs is monitored by a Web service call that is sent every 2 seconds and receives the job status. The developed sweep extension executes only one job and uploads the sequence database as well as all sub-files together (151.2 MB), only once. The single job is monitored by sending a Web service call every 2 seconds. Table 3.3 shows the analysis in detail. The CPU load was monitored for Taverna during the executions by applying the *top* command. The total packets and transfered bytes were captured by using the tool Whireshark [Orebaugh2006] applying a filter for the target registry and corresponding port. The local execution (cf. Figure 3.6 (a)) took 4.02 minutes in Taverna and the parallel execution (cf. Figure 3.6 (b)) took 1.51 minutes in Taverna (same for sweep mechanism).

Another survey was conducted and presented in [Holl2011], using the LOCUSTRA bioinformatics application [Zimmermann2008]. This workflow predicts the secondary structure of a protein, based on its amino acid sequence by using classification via Support Vector Machines (SVMs).

3.4 Discussion

This section presented an evaluation of recent workflow management systems and identified Taverna to be the most suitable one regarding the needs of this thesis investigation. The developed UNICORE-Taverna plugin showed a performance speed up against the usual execution on a client machine. This is partially based on the enhanced compute power available on supercomputers and foremost based on the possibility to decompose the execution of applications. Certainly, it is not possible for each application to be split up into several small executions, albeit it is very common that parallel versions of applications are available, such as MPI-based applications, which will at least speed up on a supercomputer.

The sweep extension presented later in this chapter showed that parallel execution may lead to much network and work load due to the high number of Web service calls for the submission and monitoring of several hundreds of jobs. Thus, an extension was developed to improve these issues by creating only one job description for a collection of jobs and by uploading identical files only once. The improved network load, machine workload and file upload performance was shown in an example using a workflow from the proteomics domain. Certainly, if files are small or only few parallel jobs are executed, the improvements might not be noticeable.

This, and also the relatively small overall speed-up is related to the problem of scheduling and queuing. A job submitted to a supercomputer is scheduled and may be queued before execution. That way, latency occurs which is not itemized and filtered out by the Taverna monitoring system. Thus, the overall runtime time of a job (with and without sweep mechanism) is mainly dependent on the scheduling and queuing on the target machine.

A general disadvantage of UNICORE is the strict usage of an X.509 certificates. This could create a barrier for domain scientists, lacking knowledge on this particular security mechanism. It would be much more intuitive for a user to only provide user-name and password. This security issue might be omitted with the new release of UNICORE 7.

3.5 Conclusion

Current trends in science already outline the large number of scientific experiments and data analysis that will emerge due to the "*exponential growth in the amount of biological data*" [Howe2008] during the next decade. Scientific workflows emerged as a successful tool to design and manage scientific experiments as well as large-scale data analysis. To simplify the usage of scientific workflows, optimization of scientific workflows is investigated within this thesis. Acceleration of workflows was identified as a prerequisite for the experimental assessment and approached within this chapter.

The workflow management system Taverna was selected as a coding basis after a general survey, due to its high number of supported life science applications and its broad expandability. To manage optimizations in a reasonable execution time, a new plugin for parallel job submission for the workflow management system Taverna was presented. The developed plugin allows users to create, share and execute applications in parallel on a high-performance computing resource. It also benefits developers, who only need to provide a few application meta-data to enable the applications use in Taverna, as the plugin automates the creation of the necessary application GUI and services.

The plugin improves the performance of the execution of activities, especially life science applications with the ability to run in parallel. In order to avoid network load and machine load due to many Web service calls as well as to improve the file upload, the proposed plugin was further extended with a recently developed sweep mechanism in UNICORE. This extension is able to collect parallel jobs and submit these in only one job description. This reduces the Web service calls for submission and monitoring and also similar file inputs have to be staged in only once. The main contributions of this chapter can be summarized as follows:

- The Taverna workflow management system is the method of choice for life scientists and will be used as a reference implementation within this thesis.
- A Taverna plugin was developed to allow parallel job execution to distributed computing resource.
- Improvement of this plugin by a newly developed feature of UNICORE. File upload rate, workload and bandwidth are reduced.
- A use case taken from the proteomics area showed that with the acceleration of single workflow applications, the overall execution time required for optimization can be reduced.

Chapter 4

A Framework for Scientific Workflow Optimization

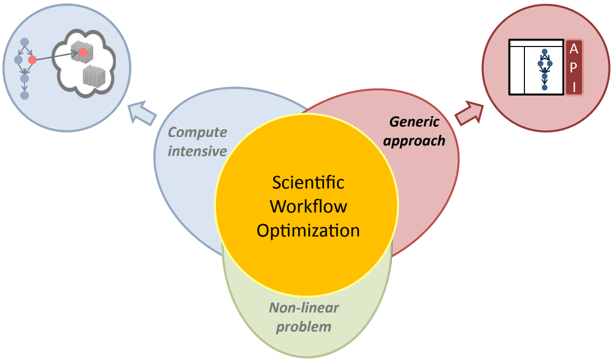


Figure 4.1: The three tier concept for workflow optimization. Second goal: generic and automated approach to be multipurpose extensible.

The optimization model for scientific workflows introduced in Chapter 2.4 can target different levels of the workflow description. Additionally, different optimization algorithms may be used for the optimization process. In order to cover many different scenarios a generic and extendable approach is investigated within this thesis. This approach should handle optimization in an automated manner with less user interaction and keep the effort of integrating new optimization algorithms small for developers.

The requirements addressed by this chapter are i) developing a theoretical approach for optimization methods and highlight the different levels of workflow optimization, ii) the implementation of these methods as a generic and automated optimization mechanism in a commonly used SWMS, and iii) an approach to move the optimization process to distributed computing infrastructures by developing a method that extends the proposed parallel execution plugin and its security mechanism.

4.1 The Approach of Scientific Workflow Optimization

The entire process of an experiment investigation by a scientific workflow is described as *scientific workflow life cycle* in literature. The workflow life cycle describes each state of the evolution of a workflow from the initial design to a final result. The common scientific workflow life cycle foresees an *analysis and learning* phase after the workflow execution [Gil2007; Ludascher2009b; Deelman2006a] in order to refine the initial design and head towards the final result, for example by changing input parameters or exchanging a specific algorithm. While for smaller workflows or a sample data set this trial and error technique may be feasible, for larger workflows and the entire data input set this mechanism becomes impractical and time consuming due to the large number of choices. Another resource-efficiency factor is the re-execution of the entire workflow, whereas typically only parts of a workflow require refinement [Wassink2009a]. To tackle these issues, this thesis proposes a new phase concatenated with the common life cycle to perform optimization in a more efficient and automated manner. The following section will shortly describe the common scientific workflow life cycle and the approach of a new optimization phase. This phase is then outlined and different levels of workflow optimization will be evolved. Finally, the definition of the optimization target required to continue the cycle, is defined.

4.1.1 A New Optimization Phase in the Scientific Workflow Life Cycle

The analysis of the State-of-the-Art workflow lifecycle and the approach of a new phase was already published in [Holl2012b] and the following sub-sections originate in this publication.

"The development of an *in silico* experiment is typically organized in a cyclic process with several steps. In literature, the term *workflow life cycle* [Aalst2004] has been shaped for this development process and several successful applications to scientific workflows have been described [Gil2007; Ludascher2009b; Deelman2006a; Ludascher2009a; Fan2011].

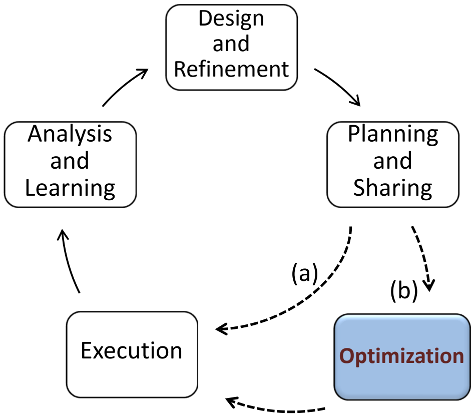


Figure 4.2: "(a) The well described scientific workflow life cycle. (b) The scientific workflow life cycle extended by the new *Optimization* phase." (Source: [Holl2012b])

The definitions of the life cycle may vary in their particular specifications but generally lead to the same architecture shown in Figure 4.2 (a). The individual phases are described in the following.

Design and Refinement Phase

The cycle usually *starts* with the design of a new or the refinement of an existing workflow taken from a repository. During this phase the components are selected, representing the individual steps of an experiment. At the same time, the composition of these components is established. This includes the precise definition of the dependencies of data and components.

Planning and Sharing Phase

Planning is turning the abstract workflow created during design phase into a concrete executable workflow. This is achieved by mapping abstract parts to concrete applications or algorithms. Parameters and data sources are defined and execution resources are selected. A thorough planning is particularly important for large-scale and compute-intensive workflows (applications) as requirements for HPC, Grid or Cloud resources have to be precisely defined in these cases. After the last cycle, this phase is used to share the designed workflow with the community in an e-Science infrastructure so that other researchers can access and then run or extend them. Depending on the used scientific workflow management system, workflow templates or workflow specifications are shared.

Workflow Execution Phase

Workflow execution is typically managed by a workflow engine. This engine maps the executable workflow to an appropriate execution environment by retrieving information about available software, computing resources and data resources and considering the proposed requirements. The workflow components are then executed in the predefined order, consuming the defined data while being monitored by the engine. The results of the workflow execution are sent back to the engine, and passed on to the user.

Analysis and Learning Phase

In order to successfully evolve the scientific experiment, the scientific workflow life cycle contains an analysis and learning step as the *last* phase. Some authors also include the publishing of the workflow and the results into this phase [Ludascher2009a; Fan2011]. The phase involves the examination of the obtained results as well as their comparison to those of other *in silico* and *in vitro* experiments.

Commonly, scientists *restart* the workflow life cycle after this analysis step, if results do not match their goals or expectations. During subsequent cycles of investigation, the workflow is adapted and tested again.

The New Optimization Phase

This thesis proposes an automated and generic process as requirement to investigate the optimization of scientific workflows. Within the common workflow life cycle, optimization is manually performed within the *Analysis and Learning* phase and consequently the entire workflow has to be re-executed in order to obtain the refined results. Thus, an arrangement of phases is required, which ensures the computationally expensive execution of the workflow not to be triggered, while intermediate results from partial workflow executions indicate that execution of the current workflow will not lead to any acceptable result. Even if the workflow involves large-scale, compute-intensive or time consuming components, another arrangement is favorable so that those components can be either ignored, approximated or optimized with special attention. Due to those reasons, this thesis proposes a new *Optimization* phase that performs automated workflow optimization before the *Workflow Execution* phase, as shown in Figure 4.2 (b).

In this phase, optimization can take only parts of a workflow into account for the adaption and re-execution. Furthermore, the new phase can describe improvements as an automated approach, i.e., it is possible to set up optimization processes, which are guided by the scientist, but hide all the complex details of the automated optimization method and execution from the user (cf. Section 4.3 and Section 4.4)." (Modified from [Holl2012b].)

In order to support the *Optimization* phase in a SWMS, different levels of workflow optimization and the design of an optimization framework is described in the following.

4.1.2 Investigation of Different Optimization Levels

A variation of the following section is described in [Holl2013f].

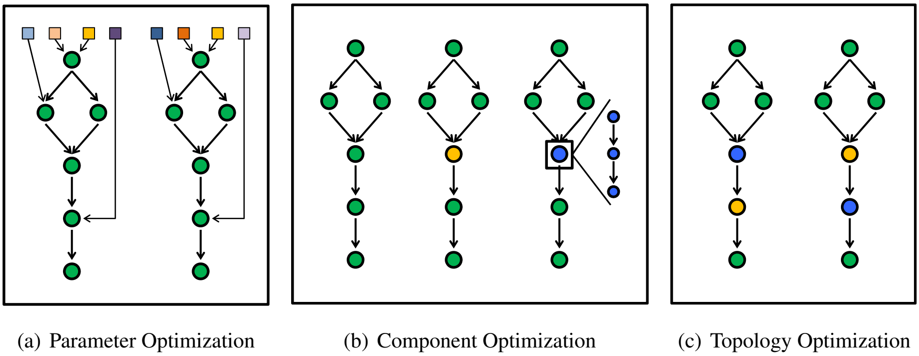
"The novel proposed scientific workflow *Optimization* phase (cf. Figure 4.2 (b)) is generally not limited to any particular type of workflow optimization. Optimizations can target different levels of the workflow description as stated in Chapter 2.4. In this thesis, three different levels of workflow optimization are investigated and defined: parameters, workflow components and workflow topology. All optimization levels aim at finding the optimal workflow set-up that optimizes the quality of the scientific result. Hence, each of these workflow optimization levels could be implemented via various optimization algorithms. For example, parameter optimization could be implemented by a Genetic Algorithm or by a Particle Swarm Optimization.

Parameter Optimization

The most common task is the improvement of the parameters of a workflow, as shown in Figure 4.3(a). Workflows may combine various data inputs and contain several different components, each with its own set of input parameters. Thus, the number of possible parameter combinations for any non-trivial workflow provides a considerable challenge for parameter optimization. Additionally, the optimal parameter combination of a workflow may depend on its input data and some parameters will have more influence on the final result than others.

In order to assist researchers in their search for optimal parameters, methods for automated parameter sweeps have been developed [Abramson2009]. However, with increasing complexity of a scientific workflow, i.e., with rising number of parameters, parameter sweeps become impractical. Therefore a method is required which handles non-linear workflow optimization in a reasonable amount of time and also needs to explore a much smaller part of the parameter search space to find good solutions. Hence, this thesis proposes the use of heuristic optimization algorithms such as Genetic Algorithms [Holland1992b] or Particle Swarm Optimization [Engelbrech2005] (for details, the reader may refer to Chapter 2.2)." (This sub-section was modified from [Holl2013f].)

However, not all parameter values in the parameter search space nor all parameter value combinations may be valid during execution. The search algorithm should be aware of



(a) Parameter Optimization (b) Component Optimization (c) Topology Optimization

Figure 4.3: The three defined levels for workflow optimization. Component optimization shows the usage of a sub-workflow as an example.

these prohibited regions as they pose challenges even for intelligent and time efficient algorithms. Certainly, the user may be aware of some of these restrictions but may not want to itemize all the specific constraints. Thus, in order to reduce the search space even further, the user should be guided through an advanced setting of the different possible constraints and dependencies between parameters. These dependent parameters can be part of only one application or belong to several applications. Dependencies could be described as mathematical or logical functions as well as a fixed set of allowed value combinations. For example, if parameter A and parameter B have a numerical dependency, an example definition is: $\forall A, B \in \mathbb{N} : |A - B| > 3$. In almost the same manner, logical dependencies could be described. Additionally, a parameter can depend on other parameter values while defining a specific value. For example, if parameter A is set to true, parameter B can only be greater than 5. And if parameter A is set to false, parameter B has to be less or equal 4. Such as $A, B = \{(a, b) \mid a = true, b > 5 \vee a = false, b \leq 4\}$. Fixed and intuitive input masks should help scientists to provide all these information.

Structural Optimization

Component optimization

"Sometimes, two or more algorithms implement equivalent functions but with different characteristics. Each of them may be optimal for different input data. Hence, component optimization, as shown in Figure 4.3(b), attempts to find the best algorithm to fulfill a given subtask.

An example component optimization is the choice of a local sequence alignment method, which is implemented among others by heuristic algorithms such as BLAST [Altschul1990], by implementations of the Smith-Waterman algorithm [Gotoh1982] or by profile alignment methods like PSI-BLAST [Altschul1997]. Thus, the user requires to know, which sequence alignment method performs best. The replacement with a *'null*-component' would also be possible for instance for filter or normalization algorithms. Components which have the same input and output data type could be switched on and off for testing their efficiency and utility.

As each algorithm and implementation has a specific parameter set, parameter optimization should be applied in combination with component optimization. In principle both optimization types can be jointly encoded for example in a Genetic Algorithm by forming sub-populations or by defining separate parts on a chromosome for each set of component specific parameters of which only one is active at a time. However, for large numbers of components with alternative methods and in general for optimization of workflows with many parameters more sophisticated techniques may be required to ensure sufficient sampling of the search space in a reasonable amount of compute time. Apart from user constraints and approximation methods, any information obtained from optimizations of similar workflows might be helpful to guide the optimization process.

Topology optimization

In some cases the result of a workflow may benefit from the reorganization of the data or enactment flow, as shown in Figure 4.3(c). Examples are cases where the execution sequence of consecutive filters, clusterings or scalings can be swapped because input and output formats are identical. The performance of the combination of a clustering method and a filter, for example, might differ depending on whether the filtering acts on the input items before the clustering step or on the cluster representatives. The result, however, would be a subset of the input items in both cases." (This sub-section was modified from [Holl2013f].)

Component and topology optimization may also include the exchange (interchange) of entire sub-workflows. Sometimes, a specific function is employed by a sub-workflow, and due to pre- or post-processing reasons this sub-workflow can only be replaced entirely by other sub-workflows or single components and vice versa.

A major requirement of component and topology transformations is additional logic, such as an ontology that enables the optimization method to determine which algorithms or sub-workflow perform equivalent functions. A similar ontology and a set of methods for data transformations are needed for input and output types.

This issue can be explained by the lack of standard in data formats. As a result, applications do typically not provide interchangeable data formats and therefore a similar ontology would be required. In order to overcome the issue of incompatible data types of the individual components, so-called *shims* might also be required. They are small adapters that transform data into the correct data type (e.g. as proposed by Radetzki et al. [Radetzki2006]).

4.1.3 Definition of the Optimization Target

In Chapter 2.2, heuristic search algorithms were identified as the preferable optimization method for scientific workflows. Due to the nature of heuristic optimization techniques, users always have to define an objective function that provides the optimization engine with a numerical metric for the quality of a specific result. In order to employ an automated scientific workflow optimization scientists have to provide a fitness function for scientific workflow results. As stated in Chapter 2, the optimization criteria can be multi- or single-objective. Depending on this, researchers have to provide a single measure for single-objective optimization and several measures with potential weights for multi-objective fitness functions. The fitness functions themselves should be encodable in any programming language or calculable by an external program to facilitate the usage. Therefore, the fitness function was designed to be represented by workflow output port(s).

The fitness measure itself is specific for the particular workflow or research goal but independent of the optimization type and the implementation in question. Consequently, a set of standard measures in terms of Taverna components may be provided to the users. Among others, examples for such standard fitness measures could be the Matthews correlation coefficient [Matthews1975] or the area under the curve (AUC) of a receiver operating characteristic (ROC) curve. The Matthews correlation coefficient is a quality measure for two class classification results. Therefore it can be used for tuning the parameter of a workflow but also for rating the result of two components. The area under the curve may serve as a function which quantifies the difference of a 'gold standard' to the simulated result [Baker2010].

Another approach could also rely on direct user feedback given to each workflow result; nevertheless this mechanism would be time inefficient and therefore require a complex interface set-up.

4.2 The Usability Compliance of Workflow Optimization

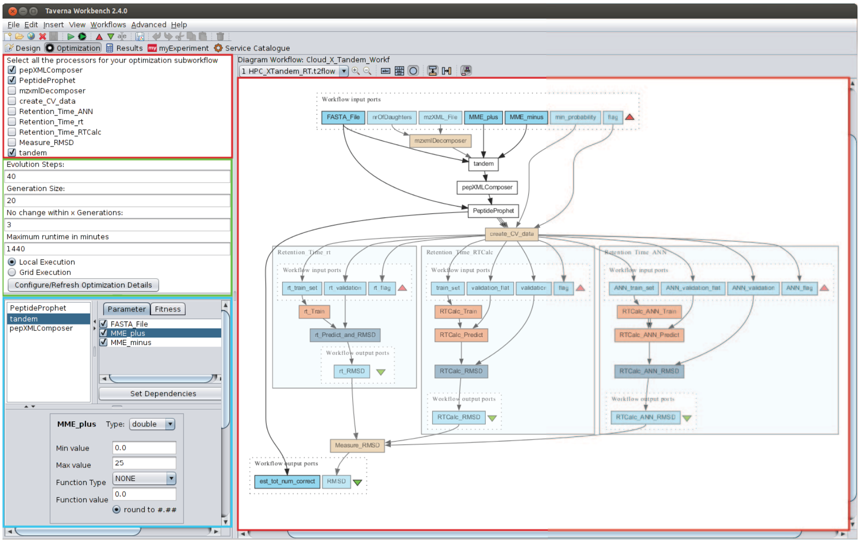
Apart from the presented workflow optimization levels it is conceivable that also other levels may be defined in the future. The same holds true for optimization algorithms, which are in a continuous development process, and former algorithms may be replaced with novel ones. Users may then demand these novel optimization techniques and therefore the effort to adopt these novel techniques should be kept as small as possible. Thus, a generic and automated framework was suggested by this thesis approach to tackle many different and future aspects of workflow optimization (cf. Figure 4.1).

In order to let developers adapt and reuse the framework, it is necessary to provide a convenient frame including a basic stock of functionalities. From the optimization logic point of view, the back-ends of optimization techniques have the same requirements, namely several sub-workflows which have to be involved, enacted, monitored and results extracted. These mechanisms rely on Taverna specific functionalities and require to be decoupled from the proper methods required for a specific optimization algorithm. Therefore they should ideally be offered by the framework to allow developers to extend new arbitrary optimization techniques with less effort.

An analogous approach is required considering the Graphical User Interface (GUI). The user desires to avoid script programming and demands a user-friendly graphical interface. This implies that the interface should use common interface elements, be less complex and have a similar look and feel for the different optimization algorithms. Especially the selection of a sub-workflow is required by all optimization techniques. In order to prevent developers from designing and developing these interfaces at each time, general required elements should again be decoupled from specific optimization interface elements. Thus, the framework must offer general GUI elements to easily extend an optimization algorithm specific GUI.

A basic accessory aspect for ordinary usage of optimization methods concerns the complexity of workflows, which should not increase during the optimization process. Users may not have to arrange the workflow within another workflow, which performs the essence optimization process. This workflow in a workflow approach was used by other related work (cf. Chapter 2 [Crick2009] and [Abramson2010]) and was identified as not comfortable and user-unfriendly. The presented framework needs to manage this complex structure and invocation in the background, hidden from the user.

In order to approach these requirements and apply a user and developer compliant framework, a versatile Application Programming Interface (API) is required. This interface should facilitate the extension for developers and usage for scientists. The design and implementation to tackle model extensibility and a graphical user interface is described in the form of a novel optimization framework in the following section.



Select all the processors for your optimization subworkflow
☑ pepXMLComposer
☑ PeptideProphet
☐ mzxmlDecomposer
☐ create_CV_data
☐ Retention_Time_ANN
☐ Retention_Time_rt
☐ Retention_Time_RTCalc
☐ Measure_RMSD
☑ tandem

Evolution Steps: 40
Generation Size: 20
No change within x Generations: 3
Maximum runtime in minutes 1440
● Local Execution
○ Grid Execution

PeptideProphet
tandem
pepXMLComposer

☑ FASTA_File
☑ MME_plus
☑ MME_minus

MME_plus Type: double
Min value 0.0
Max value 25
Function Type NONE
Function value 0.0
● round to #.##

Figure 4.4: The Taverna optimization perspective implemented by the framework (red), optional panel by the framework (green) and implemented by a plugin (light blue).

4.3 The Taverna Optimization Framework

The workflow management system Taverna [Missier2010] was selected as reference implementation in this thesis (cf. Chapter 3.1), thus the framework for scientific workflow optimization is being integrated in Taverna. The architecture of this concept is shown in Figure 4.5. The main requirements of the framework were detailed in the last section as:

- Hide complexity from the user.
- Provide extensible and user-friendly graphical interface.
- Implement an extensible method that provides generally required functionalities.

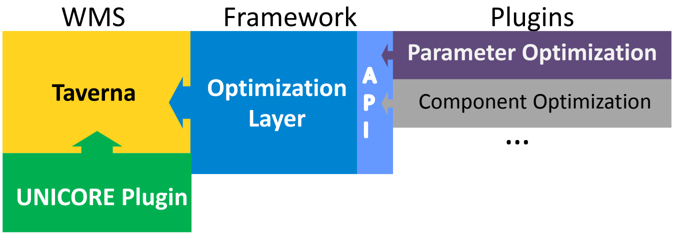


Figure 4.5: Architecture of the Taverna Workflow Management System, with the previously described UNICORE plugin (cf. Chapter 3), and the proposed extended optimization framework, including the API (cf. Chapter 4.3) as well as possible optimization plugins (cf. Chapter 5.2).

Summarizing this, the framework must offer an API in order to provide access to Taverna specific functionalities for model and GUI extensions. However, Taverna was not dedicated to support scientific workflow optimization methods and thus, a specific approach had to be developed for the integration. This approach is described in the following and includes the architecture and implementation of the framework, which was also described in [Holl2013f].

"To integrate workflow optimization within Taverna and provide Taverna specific functionalities, a major requirement is the accessibility of all input and output parameters and also the sub-workflow. The accessibility must be guaranteed, as the purpose of workflow optimization is to modify the workflow parameter or workflow structure, execute the modified sub-workflows and evaluate the results. Furthermore, a mechanism is required to i) interrupt the execution of a workflow before the entire workflow is executed, and ii) execute a sub-workflow several times for optimization purposes.

In order to implement these requirements and receive access to the Taverna execution model and graphical user interface, the first approach aims at extending and reusing Service Provider Interfaces (SPIs), which are provided by the Taverna system. One such SPI provides usage of the processor dispatch stack of Taverna as described in [Missier2010] and shown in Figure 4.6. The processor dispatch stack is a mechanism to manage the execution of a specific activity. Before the activity is invoked, a predefined stack of layers is called from top to bottom and after the execution from bottom to top. Each layer implements a special functionality which is important for the activity invocation. In order to integrate the optimization execution into Taverna, this dispatch stack was extended with

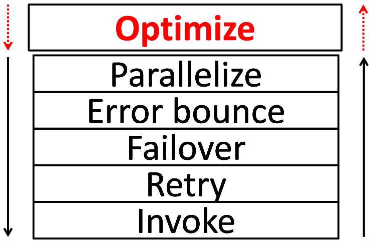


Figure 4.6: The common Taverna dispatch stack in black and the extended optimization layer in red.

a new *optimize* layer on top of the stack, see Figure 4.6. This allows to interrupt the process of workflow execution before a specific component is executed.

Additionally, advantage was taken of the sub-workflow concept, provided by Taverna. It allows for the definition and execution of a sub-workflow. This implies that one can manually select a sub-workflow, which is interpreted by Taverna as a single activity and therefore processed by the dispatch stack. Only at the lowest layer the sub-workflow is decomposed into the individual activities, each of which again traverses the stack itself. This fact was utilized by defining those parts of the workflow that shall be optimized as a Taverna sub-workflow.

Leveraging these two concepts, the processor dispatch stack provides access to the sub-workflow, which in turn provides access to all parameters, data structures, data flows etc. during the invocation of the *optimize* layer, please refer to Figure 4.7 for more details.

The optimization framework also utilizes a GUI SPI and implements a new uniform perspective into the Taverna Workbench in order to implement a general set-up of an optimization run, as shown in Figure 4.4. In the new integrated perspective, the common workflow diagram (cf. right hand side red box of Figure 4.4) and a selection pane (cf. left hand side red box of Figure 4.4) were arranged. Within these, the user can define the sub-workflow, which shall be subject to the optimization. After the selection, all components which are not subject to optimization, appear greyed out in the workflow diagram. Internally a new sub-workflow is created. The pane at the lower left shows a specific interface implemented by the respective optimization plugin (cf. blue box in Figure 4.4).

In order to provide access and usage of the execution model and user interface mechanisms, an API was developed for the framework. This API let developers extend op-

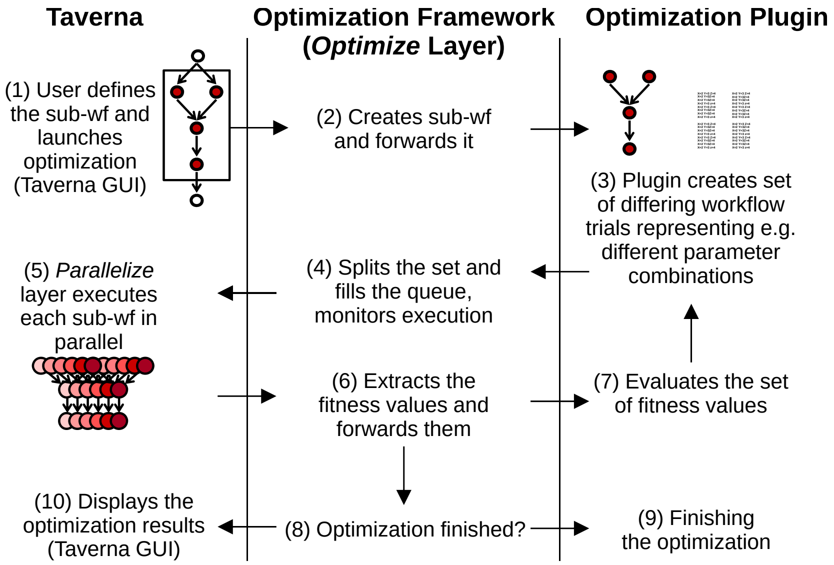

Taverna
Optimization Framework
(*Optimize* Layer)
Optimization Plugin
(1) User defines the sub-wf and launches optimization (Taverna GUI)
(2) Creates sub-wf and forwards it
(3) Plugin creates set of differing workflow trials representing e.g. different parameter combinations
(4) Splits the set and fills the queue, monitors execution
(5) *Parallelize* layer executes each sub-wf in parallel
(6) Extracts the fitness values and forwards them
(7) Evaluates the set of fitness values
(8) Optimization finished?
(9) Finishing the optimization
(10) Displays the optimization results (Taverna GUI)

Figure 4.7: The control flow of the new optimize layer and a parameter optimization plugin for workflow optimization. The colored activities in phase (3) define one sub-workflow. Before and after the optimization the workflow is regularly executed. ('sub-wf' stands for sub-workflow) (Source: [Holl2012b])

timization methods without reinventing and reimplementing Taverna specific execution and GUI functionalities. The API provides methods for i) requesting the GUI pane to provide specific text fields, selection options, etc. for the particular optimization method, ii) starting the optimization process of the specific plugin, iii) receiving the modified parameter or sub-workflows to start the execution, iv) receiving the fitness value from the workflow to forward it to the plugin, v) requesting the termination criteria and decide whether the optimization is finished, vi) requesting the best result to return it to the user. The employment of this API is described in the following (cf. Figure 4.7) and an example usage is described in Chapter 5.2.

In the new optimization perspective, which is provided by the framework, the user can select the sub-workflow and optimization specific information (cf. Figure 4.7(1)). After the user has provided these required inputs, the newly integrated run button activates the fully automated optimization process. The workflow is executed until the respective sub-workflow is reached. Then the new *optimize* layer extracts required information and forwards it to the particular optimization plugin for modification (cf. Figure 4.7(2)). The proper optimization method and sub-workflow refinement is then handled by the respective optimization plugin and not by the framework. The optimization plugin returns a set of new sub-workflow entities, each with a different set of parameters (cf. Figure 4.7(3)). The optimization framework executes this sub-workflow set by utilizing the topmost Taverna standard layer, namely *parallelize* (cf. Figure 4.7(4)). More precisely, it provides a queue filled with sub-workflows for the *parallelize* layer, which in turn pushes each sub-workflow down the stack in a separate thread. This triggers the parallel execution of all sub-workflows (cf. Figure 4.7(5)). After execution of the sub-workflows, the *optimize* layer receives a set of results from the *parallelize* layer. This result set, which represents the fitness value, is again passed through to the specific optimization plugin for analysis and evaluation (cf. Figure 4.7(6 and 7)). If the optimization has reached the termination condition, the optimization is finished (cf. Figure 4.7(8 and 9)). Otherwise, the plugin creates another set of sub-workflows for the next round of sub-workflow executions (cf. Figure 4.7(3)). If any termination condition is reached, the optimal sub-workflow and parameter set is returned to the user and the workflow execution can be resumed." (Modified from [Holl2013f].) Finally, the result is presented to the user (cf. Figure 4.7(10)), who can store it as a file including statistics and meta-data of the optimization process or as a Taverna conform workflow input file for sharing purposes.

4.4 Enabling Optimization on Distributed Computing Infrastructures

The proposed scientific workflow optimization framework schedules several independent executions of the sub-workflow instances. As scientific workflows can become very complex [Barga2007], the workflow executions should run in parallel to perform the optimization in a reasonable amount of time. Nevertheless, running several parallel workflow executions may causes a bottleneck in the workflow engine [Gorlach2011] and is not recommended on a Taverna client in many cases by means of a high client machine workload. A shift of the parallel workflow executions to distributed computing resources may lower the workload on a client and is approached within this section. The design and implementation of this improved parallel workflow execution support was already published in [Holl2012a] and is described in the following.

4.4.1 Three Tier Execution Architecture

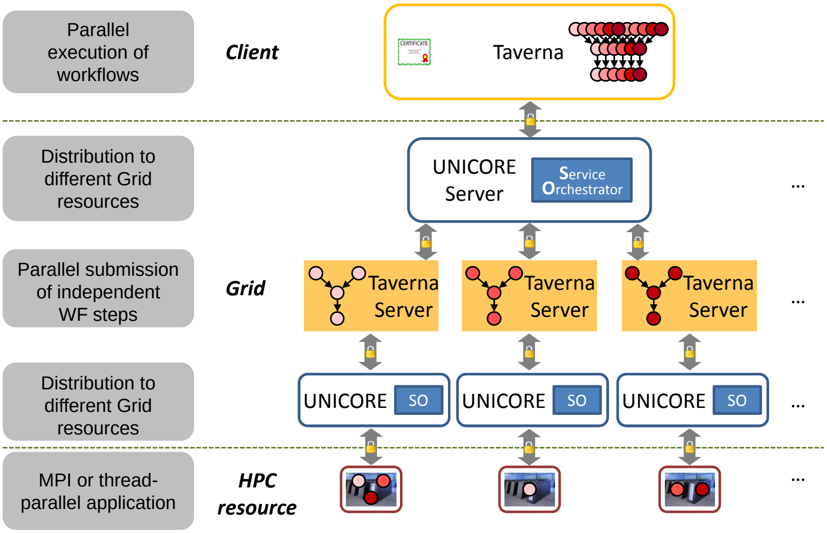

Parallel execution of workflows
Client
Taverna
CERTIFICATE
Distribution to different Grid resources
UNICORE Server
Service Orchestrator
...
Parallel submission of independent WF steps
Grid
Taverna Server
Taverna Server
Taverna Server
...
Distribution to different Grid resources
UNICORE
SO
UNICORE
SO
UNICORE
SO
...
MPI or thread-parallel application
HPC resource
...

Figure 4.8: The three tier execution architecture. The first tier shows the Taverna Workbench on the client, the second tier includes the Taverna server and the third tier consists of parallel or MPI applications. (Source: [Holl2012a])

The following section was already published and originates from [Holl2012a]. "In order to implement optimization on the Grid, the parallel process of workflow optimization is embedded in a three tier execution architecture, shown in Figure 4.8. The architecture can utilize even large computing resources as it implements a hierarchy with up to three levels of parallel execution in its architecture. At the uppermost layer, the Taverna Workbench is established on the client. In the Workbench users can set up the workflow and configure inputs and outputs as well as describe the optimization composition. As described before in Section 4.3, several workflow instances are created in each generation of the optimization process. In order to allow workflow execution in a distributed fashion, the UNICORE-Taverna plugin described in Chapter 3 was adapted to the framework. Because of the fact that all workflow executions are independent, the execution can be done in parallel, submitting each individual workflow instance to the UNICORE Service Orchestrator (SO). The SO manages the distribution of the workflow on the computing infrastructure and monitors the status of the workflow executions. For distribution it implements various brokering strategies to find the best suited set of resources and enables load balancing. By using this feature, the workflows are submitted in a distributed way and one instance of Taverna server is utilized, to execute a single workflow, as shown in the Figure 4.8 as second layer. At the second layer, each workflow itself may contain several independent steps that can be represented as individual UNICORE jobs and thus be submitted for parallel execution on High-Performance Computing (HPC) resources due to the parallel execution model in the Taverna and the job distribution of the SO.

Finally, at the third tier in Figure 4.8, some of the workflow steps may themselves be implemented as MPI- or thread-parallel applications, which constitutes the third level of parallel execution." (Modified from [Holl2012a].)

4.4.2 Implementation of Parallel Workflow Execution

The described three tier architecture can be set up with existing software components, namely Taverna components and UNICORE components. However, the highly recommended security standards obtained within distributed computing infrastructures cannot be met with this set-up. "The security issue arises at the Taverna server located at the second layer in the architecture (Figure 4.8). Here the execution of each workflow takes place. As described before, the workflows themselves may contain several Grid activities, which require to be submitted to the infrastructure. For these job submissions, the certificate of the user is required. This is due to the high security standards existing in distributed computing infrastructures [Benedyczak2011]. Usually, the execution of the workflow takes place on the client machine, where the user's certificate is available. Then the user is allowed to send signed jobs to the distributed computing infrastructure for execution. In the present case, the Taverna server has to submit these jobs to the distributed computing infrastructure. Unfortunately, the server is not granted access to the target systems nor the user certificate is available at the server. This is based on the fact that the private user certificate should never be stored on an external storage device and must be kept confidential. Furthermore, it is not secure to send the private user certificate to the server via the network. Consequently, the user has to delegate his/her access rights to the server in a different way.

In order to equip the server with these delegation rights, a new mechanism based on SAML assertion was investigated in the Taverna Workbench as well as in the Taverna server. The SAML-based mechanism is based on trust delegation utilizing the user certificate and the certificate of the Taverna server on the client. In detail, the server certificates

Distinguished Name (DN) is stored in the incarnation database (IDB) of the UNICORE server. During the request of the SO, which collects target site specific information such as available applications, it also fetches the DN of the server, as shown in Figure 4.9(1). This information is provided to the Taverna Workbench, which extracts and stores the DN. This stored DN in combination with the user's X.509 certificate is taken to create a trust delegation which delegates the user's credentials to the server (cf. Figure 4.9(2)). This delegation is sent as a file to the Taverna server with the workflow job (cf. 4.9(3)). The server can in turn use this trust delegation to create a SAML assertion. With the SAML assertion the server can submit the individual application job on behalf of the user to the target HPC system (cf. 4.9(4))." (Modified from [Holl2012a].)

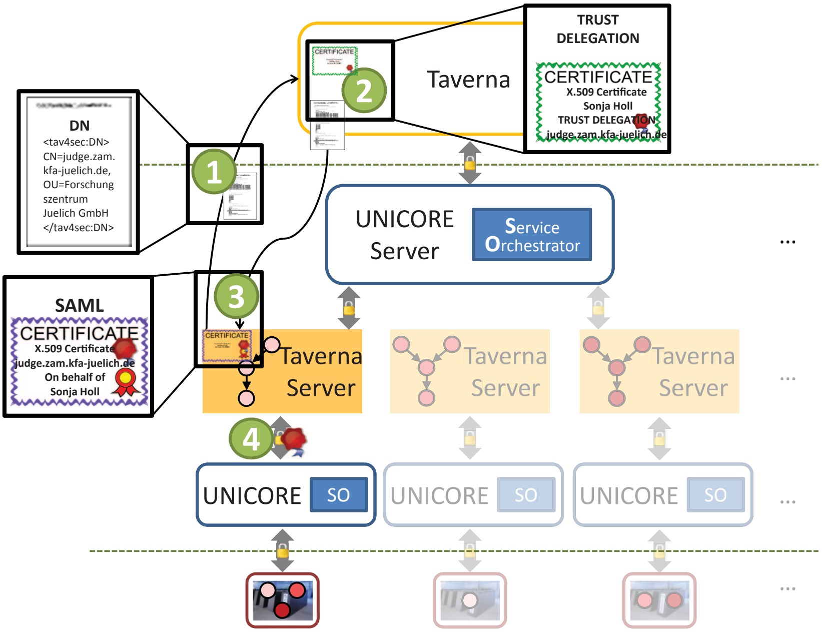
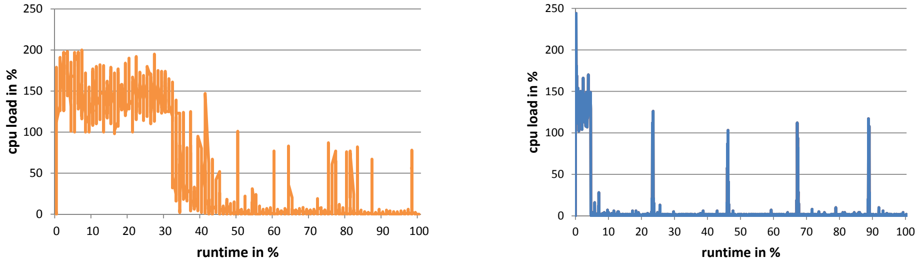


Figure 4.9: The new security propagation mechanism. In (1) the DN of the Taverna server is published via the IDB to the Taverna Workbench. Taverna produces a trust delegation in (2). In (3) the trust delegation is forwarded to the Taverna server, which in turn uses the trust delegation to assign a SAML assertion for job submission. The job is submitted by the Taverna server in (4).

4.4.3 Parallel Optimization Use Case

In order to assess the establishment of the architecture and novel security mechanism, an exemplary workflow containing three activities of which two activities should be optimized, was designed. The sub-workflow, which was defined to contain two activities, was automatically created by the optimization framework as described in the previous section. Performing the optimization, as example set-up 20 sub-workflow instances were initialized and executed in parallel, 5 times in series. For analysis purposes, the sub-workflow instances were executed in parallel on the client machine as shown in Figure 4.10 (a) and in parallel on the computing resource as shown in Figure 4.10 (b). For the parallel execution on the distributed computing infrastructure, the improved parallel execution mechanism as described in Chapter 3.2.5 was applied. A more detailed analysis of all execution scenarios and more use cases are presented in [Holl2012a] and [Memon2013].



(a) Client workload during execution of 20 workflows in parallel on the client machine.

(b) Client work load during execution of 20 workflows in parallel on the Grid. The five peaks represent the five execution series.

Figure 4.10: Client workload of two different scenarios during workflow optimization. (Source: [Memon2013])

4.5 Discussion

A generic framework for scientific workflow optimization was described in this chapter. By designing the framework in a very generic way and by providing an API, it is extensible and not limited to any optimization level or strategy. It was implemented in Taverna, a widely adopted workflow management system, to lower the hurdle for scientists for dealing with new technologies. The framework was extended with a new service for moving all optimization specific executions to distributed computing infrastructures. Using this, it was shown that the workload on the client machine was reduced. This mechanism requires access to high-performance computing resources and is therefore not recommended while executing for instance Web service based workflows. Access to high-performance resources is of necessity, too. The architectural set-up can be designed more meaningful; however the Taverna server instances are currently not able to execute several workflows in parallel. With the development of Taverna 3, this scenario could be improved as the revision of the server is planned (Details gathered from [UoM2009]).

Compared to other design frameworks from the engineering domain, the presented approach can accomplish some of the recent faults identified in most of the current multidisciplinary design optimization frameworks, namely "*User guides together with wizards, project templates and GUI to aid the optimization process. Parallel and distributed computing support. Domain-Specific Languages for objective function and constraints formulation*" [Parejo2012]. The framework provides an easy and intuitive user interface to make optimization usable for computer experts and non-experts alike. Grid support and developed parallel execution allow for the efficient calculation on heterogeneous and distributed computing resource. Based on Taverna's multipurpose approach, the fitness function can be described by means of any conceivable programming language.

4.6 Conclusion

The common strategies for workflow improvements such as trial and error or parameter sweeps are becoming impractical and inefficient for complex life science workflows. Therefore the optimization of scientific workflows was identified as a crucial and novel research field to assist users during the set-up of *in silico* experiments [Abramson2010]. Within this chapter, workflow optimization was integrated into the common scientific workflow lifecycle by adding a new *Optimization* phase between *Planning and Sharing* and *Workflow Execution*. Three levels of workflow optimization (parameter, component and topology) were introduced and described. Plugins implementing these levels and other optimization algorithms should be independently and easily pluggable as well as usable in Taverna. Therefore an automated and generic optimization framework was developed and described within this chapter in order to offer scientists a comprehensive tool for achieving optimized scientific results. The framework was integrated into Taverna, a popular SWMS. It was equipped with basic functionalities for workflow optimization and it also offers an application programming interface to be extended by various optimization plugins. Developers benefit from this design as they do not have to deal with any Taverna specificities, model executions or GUI implementations. The framework offers a generic platform to quickly extend novel plugins with less effort. Scientists benefit from this design as new algorithms may be promptly available without installation and familiarization with a new tool.

In a following step, the framework was equipped with a service that allows for moving parallel workflow executions to distributed computing resources. By implementing this, the workload on the client machine was lowered to an acceptable degree, as the entire optimization process can be conducted on a distributed computing infrastructure. The implementation was integrated within a three tier parallel execution architecture and thus enables high utilization of distributed computing resources and provides a significant speedup of the optimization process. The issue to conform to the UNICORE security standards was solved by implementing a mechanism using a trust delegation on the client and a SAML assertion on a Taverna server for submission to the distributed computing infrastructure.

The main contributions of this chapter can be concluded as follows:

- Definition of a new *Optimization* phase and three different workflow optimization levels.
- Development of an optimization framework with API that extends Taverna and implements specific mechanisms for optimization execution and Graphical User Interface for an eased extension by developers and usage by scientists.
- Equipment of the framework with a service that allows for moving parallel workflow executions to distributed computing infrastructures to minimize the load on the client.

Chapter 5

Optimization Techniques for Scientific Workflow Optimization

In the last chapter, a generic and automated optimization framework was introduced in order to fulfill the second requirement raised to assess scientific workflow optimization. The framework offers an application Programming Interface (API) for the extension of different optimization levels or algorithms. Different optimization levels were determined and presented, namely parameter, component and topology optimization. In this chapter, the non-linear problem of workflow optimization (cf. Figure 5.1) is approached by proposing an exemplary plugin, which apply a meta-heuristic optimization algorithm. The plugin primary implements for parameter optimization and leveraging the proposed API of the optimization framework. As the developed framework is extendable by any combination of optimization algorithm and level, this chapter shows one possibility of an optimization extension that can be engineered by developers. The second part of this chapter investigates different use cases taken from the life science domain to test workflow optimization. On the one hand, parameter optimization is taken into account, and on the other hand, higher level structure optimization use cases will be validated.

5.1 Optimization Techniques for Scientific Workflow Parameters

"Life science workflows may combine various data and configuration inputs and contain several different applications. Typically, each component has a number of parameters and

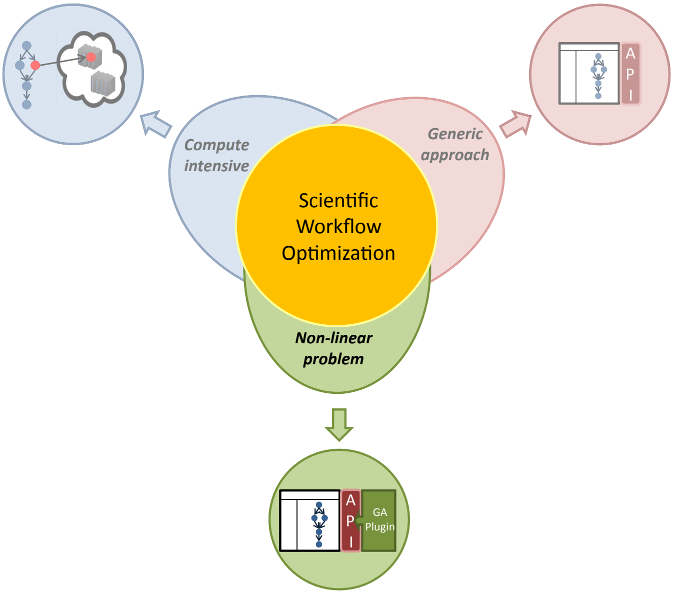


Figure 5.1: The three tier concept for workflow optimization. Third goal: addressing the non-linear optimization problem.

the quality of the final result critically depends on good choices for them. Additionally, data inputs and configuration parameters may have different influences on the final result. Thus, the combinatorial number of possibilities for parameter values becomes a challenge for researchers." [Holl2012b]. Whereas recently only single applications (or scripts) and their specific scope have been optimized in the life science domain [Handl2007; Sun2012; Unger2004; Gondro2007], configuration parameters are still calibrated by using the default values, trial and error or parameter sweeps. Due to this, workflow parameters were determined to be the most important optimization target and therefore particular attention will be given to workflow parameter within this thesis study. For a detailed description of parameter optimization, please refer to Chapter 4.1.2.

In order to optimize the parameters of a scientific workflow, a measure for the workflow result is required. This measure will be evaluated (minimized or maximized) regarding a numerical value and is called *objective function*. Typically, the objective function is calculated by a standard performance measure or a simulation software [Deb2001]. In the life science domain, these software programs are usually NP-hard problems [Bader2004].

Scientific workflows aggregate such programs by linking them together and are thus NP-hard combinatorial optimization problems.

In Chapter 2 different optimization algorithms have been presented. It was identified, that deterministic algorithms, stochastic and heuristic methods can be used to solve non-linear optimization problems. Genetic Algorithms and other metaheuristic optimization methods need a significant lower number of explored solutions to find a potential good one. Due to the considerable time required for the execution of a scientific workflow compared to the execution of a single program, the parallel execution of optimization was imposed as a requirement and implemented in the previous chapter. Heuristic methods [Pham2000; Talbi2009] perform especially well within distributed and parallel execution environments such as EAs [Back1996], in particular Genetic Algorithms (GA) [Holland1992b], Particle Swarm Optimizations (PSO) [Engelbrech2005] or Ant Colony Optimizations(ACO) [Dorigo1999] (For details, please refer to Chapter 2). Within the context of scientific workflows, different set-ups for the optimization algorithms and their entities are presumable:

- GA: Each parameter is represented by one gene and one parameter set is represented by one chromosome. Constraints give ranges to the possible values for genes. A population represents one generation of several different workflow set-ups.

- PSO: Each parameter spans one dimension of the search space and one particle represents one parameter set. Constraints define the range of their respective dimension. One iteration represents one generation of several different workflow set-ups.

- ACO: Each set of parameters is defined by one ant and each visited *node* represents one parameter. Constraints define ranges to the pheromone intensities. One iteration represents one generation of several different workflow set-ups.

In order to obtain a reasonable result in a reasonable amount of time (cf. Chapter 2.4) for the optimization, scientific workflows in the life science domain require the parallel execution of samples. As Evolutionary Algorithms (EAs) emerged as the parallel meta-heuristic of preference [Alba2013], this thesis based the example plugin implementation of workflow parameter optimization on GAs [Holland1992a]. Other decisive factors are their simplicity, proven performance, versatility and recent success in the life sciences [Niazi2012]. A significant advantage of applying an algorithm that has been in use for a long time, is that a range of mature, feature rich and well-tested libraries for GAs are available as open

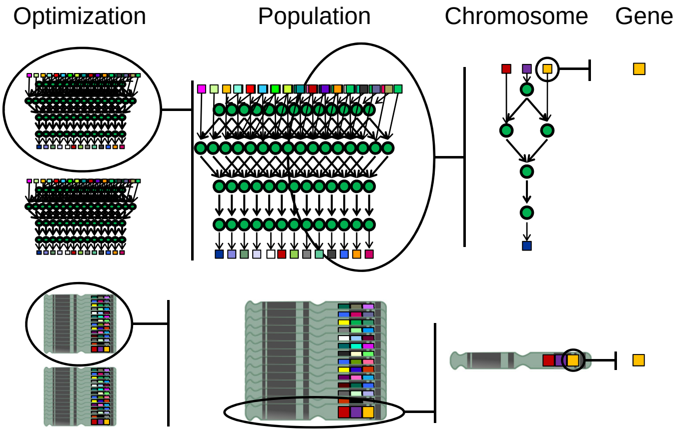


Figure 5.2: The encoding of a workflow to Genetic Algorithm entities. One parameter is represented by one gene. One workflow set-up is represented by one chromosome. One set of workflows is represented as a population.

source. This section will introduce some details about Genetic Algorithms, elaborate on the selected GA library and go on to with a possible mapping of scientific workflows to GA entities.

5.1.1 Genetic Algorithms

Genetic Algorithms (GAs) are among the most popular heuristic optimization methods and have been successfully used for more than 35 years to solve complex scientific optimization and search problems. Even in the life sciences, GAs have been successfully utilized to predict protein structures [Unger2004], calculate multiple sequence alignment [Gondro2007] or parameter estimation in kinetic models [Baker2010]. GAs do not require any derivatives and can encode both numerical and non-numerical problems. They mimic the mechanism of natural evolution by applying recombination, mutation and selection to a population of parameter vectors. Like Charles Darwin laid down the principle of the *survival of the fittest*, a competition of individuals takes place in each consecutive generation for solving a problem. Each of those generations consists of a population of individuals, which represent a point, i.e., a possible solution in the search space. An individual is analogous to a chromosome and consists of different genes. The evolution of the population of potential solutions is guided by a problem specific objective function called *fitness function* and each individual is evaluated by employing this fitness function. The fitness value determines the probability of a solution to be inherited to the next generation. A new population is created by selecting the fittest parents of the prior population; applying crossover, which represents mating between individuals, and mutating individuals, which means introducing random modifications to genes. For the selection, crossover and mutation different operators exist. For more details the reader has several sources, such as [Deb2001, pp. 88-152] or [Sivanandam2007, pp.46-72]. This search process combines exploration and hill climbing and allows the algorithm to sample many local minima in order to find the global optimum without being trapped. The GA terminates either after a fixed number of iterations has been carried out or if some fitness based threshold has been reached and the fittest individual has been selected as the final solution. In this thesis approach, this individual represents the best parameter set for a given workflow instance.

5.1.2 A Genetic Algorithm for Scientific Workflows

In order to optimize the parameters of a scientific workflow by Genetic Algorithms (GAs), a mapping between the genetic entities and the workflow is required. In this thesis, a real-coded GA was selected due to its lower computational complexity compared to binary-coded GAs [Deb2001, p. 111]. Thus, the workflow parameters were straightly encoded as genes on a chromosome as visualized in Figure 5.2. Hence, each chromosome encodes one set of input parameters and represents a particular combination of input values. One population corresponds to a set of workflow trials - several workflow instances each of which is instantiated by one chromosome.

Many GA libraries in a multitude of different languages have been developed and after evaluation of several parallel and open source GA implementations in Java, the Java-based library JGAP [Meffert2011] (Java Genetic Algorithms and genetic programming package) has been selected for this thesis project. Among others, the library ECJ [White2012] and Watchmaker framework [Dyer2010] were also taken into account. ECJ is the most popular library but cannot straightforward encode different types of real-parameters within one chromosome. The Watchmaker framework is nowadays an efficient but still novel framework; many features have just recently been developed by only a single person.

The following section describes the application of JGAP to workflow optimization and originates from [Holl2012b].

"JGAP [Meffert2011] is an open-source GA library developed by various free lancing developers, mainly by Klaus Meffert. It offers a general mechanism for genetic evolution that can be adapted to the particular optimization problem. For workflow optimization, mainly the JGAP's parallel execution mechanism and the breeder mechanism was refined. In order to allow for support of constrained parameter optimization, use was made of a special genotype in JGAP, called *MapGene*.

MapGenes can be used to define a fixed set of parameter values, from which the breeder can select only one. Additionally, a novel mechanism to add functions to numerical genes for mapping gene values to parameter values was invented: $y = f(x)$ (where y defines the parameter value, and x the gene value). One of the major changes to JGAP concerns the central breeder method. The refined breeder acts as follows: The generated population of sub-workflows is first executed and the calculated are translated into fitness values in order to determine the fittest chromosome. Then a new population is generated by applying an *a priori* chosen natural selection mechanism, for instance tournament or best chromosome selection on the old population. To these parents, crossover and mutation operators are applied and they are added to the new population. For crossover, a new blend crossover (BLX) mechanism [Eshelman1993] was implemented and extended in JGAP. BLX is a preferably used crossover method in real-coded GAs. Mutation is performed by a uniform mutation operator, which was also implemented and added to JGAP. To fill the population to the predefined population size, randomly created chromosomes are added, regarding the *random-immigrant* strategy [Cobb1993]. This procedure was selected to have a balanced relation of exploration and exploitation. Now the evolution cycle is continued with the new population.

The parallel execution mechanism of JGAP was reimplemented and is able to determine the fitness values of the chromosomes in parallel. In combination with the framework, proposed in Chapter 4.4, the JGAP library suspends and waits until all sub-workflows have been executed and the fitness values have been extracted and returned." (Modified from [Holl2012b].)

5.2 The Parameter Optimization Plugin

In order to allow automated parameter optimization of workflows from within Taverna, this section describes the approach to accomplish a Genetic Algorithm based optimization plugin that employs the developed optimization framework API described in Chapter 4. By utilizing the generic framework, developers can ignore all the Taverna specific internals such as security, execution or data handling on the Grid as well as GUI design. For

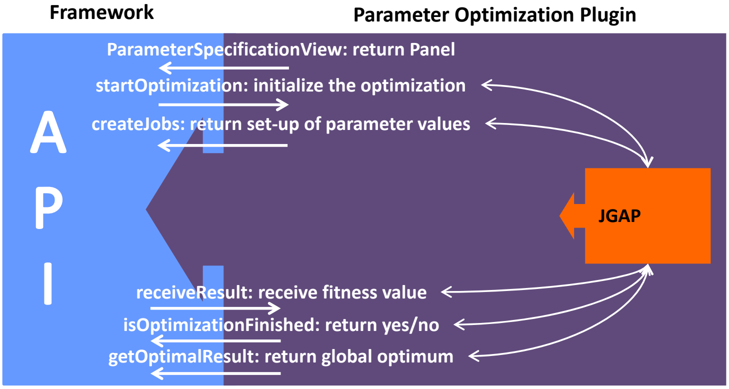

Framework
Parameter Optimization Plugin
A
P
I
ParameterSpecificationView: return Panel
startOptimization: initialize the optimization
createJobs: return set-up of parameter values
JGAP
receiveResult: receive fitness value
isOptimizationFinished: return yes/no
getOptimalResult: return global optimum

Figure 5.3: The abstract programming interface in detail, including all required methods that need to be implemented by the plugin. As example, the parameter optimization plugin with JGAP is shown.

the architecture and development, different requirements have to be taken into account according to the API description (cf. Chapter 4.3): i) provide a panel for optimization configuration, ii) provide methods to start the optimization process, iii) provide methods to return optimization samples, iv) provide methods for receiving a result, v) provide termination criteria and status method, and vi) offer a method to request the final result. In the following the design and implementation of these requirements are described.

5.2.1 Development of the Parameter Optimization Plugin

The parameter optimization plugin is plugged into the framework API by reusing the predefined interface methods. Figure 5.3 shows the interactions of the framework and the plugin in an abstract manner. By employing the previously described generic framework, developers do not have to deal with many Taverna specific internals such as execution or data handling. All accessible parameters of the sub-workflow can be included in the parameter optimization process. To allow scientists the modification of the parameter sampling space, the developed parameter optimization plugin integrates a new pane into the optimization perspective of the Taverna Workbench (Figure 5.5 (3)). In order to benefit from user knowledge as described in Section 4.1.2, the user can select which parameters shall be subject to the optimization. Optionally, a type dependent set of properties such as upper and lower bounds and a mathematical function: $y = f(x)$ (where y defines the

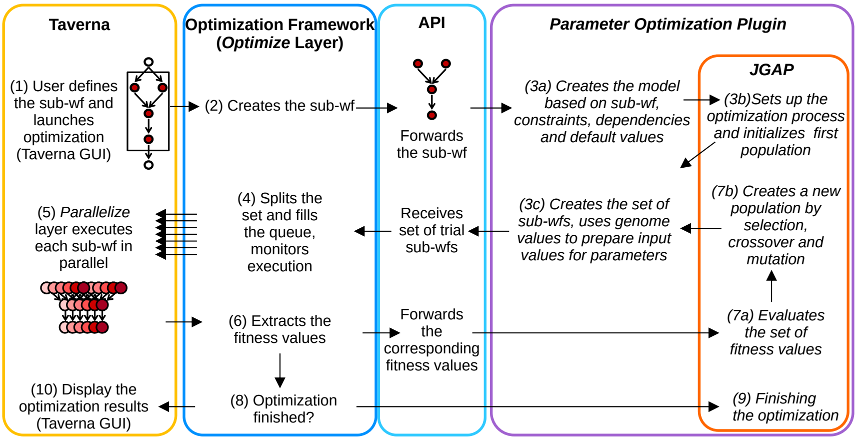
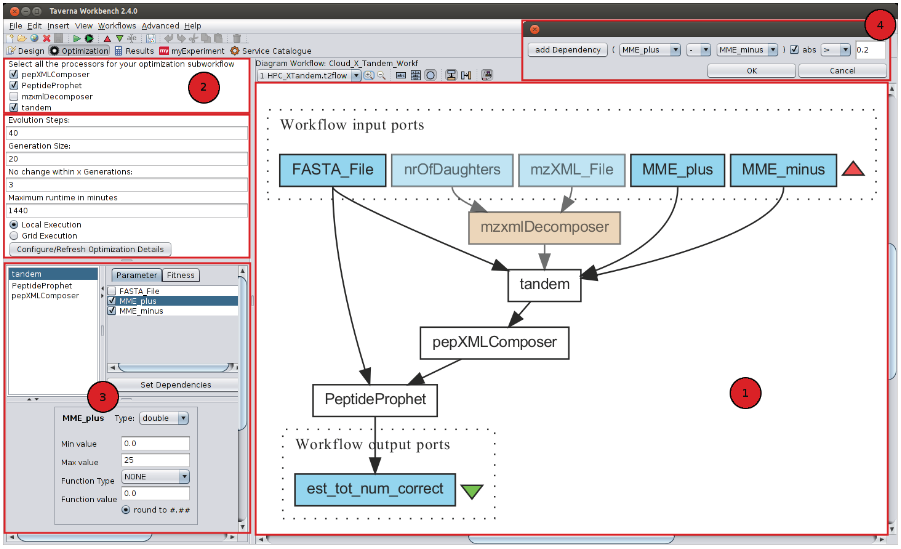

Taverna
Optimization Framework (*Optimize* Layer)
API
Parameter Optimization Plugin
JGAP
(1) User defines the sub-wf and launches optimization (Taverna GUI)
(2) Creates the sub-wf
Forwards the sub-wf
(3a) Creates the model based on sub-wf, constraints, dependencies and default values
(3b)Sets up the optimization process and initializes first population
(5) *Parallelize* layer executes each sub-wf in parallel
(4) Splits the set and fills the queue, monitors execution
Receives set of trial sub-wfs
(3c) Creates the set of sub-wfs, uses genome values to prepare input values for parameters
(7b) Creates a new population by selection, crossover and mutation
(6) Extracts the fitness values
Forwards the corresponding fitness values
(7a) Evaluates the set of fitness values
(10) Display the optimization results (Taverna GUI)
(8) Optimization finished?
(9) Finishing the optimization

Figure 5.4: The control flow of the new optimization framework and its API for workflow optimization as well as a GA-based parameter optimization plugin. ('sub-wf' stands for sub-workflow)

parameter value, and *x* the gene value) can be specified for each parameter. For discrete parameter types, the user can provide a list of valid values. It is also possible to specify dependencies between parameters either as a fixed list of allowed combinations, or as a mathematical or logical formula. The only non-optional part of the set-up process is the description of the desired optimum in terms of a *fitness function*. In the most simple case, one output of the workflow can be chosen as fitness value to be maximized or minimized. Alternatively, several outputs can be combined in a script to calculate a single fitness value. Finally, the user can configure the termination condition for the evolutionary process.

The JGAP library is completely incorporated into the designed plugin as a Java package. It performs the general optimization process, namely generation of a population, selection, crossover and mutation. The process of parameter optimization is shown as a control flow in Figure 5.4, and described in the following.

"The user initiates the optimization by selecting a sub-workflow for optimization, defining parameters as well as constraints and finally providing input values for the remaining non-optimized parameters. During the optimization run, the optimization framework invokes the parameter optimization plugin and forwards the user-selected sub-workflow cf. Figure 5.4(2). The plugin initializes a new evolutionary process by using the extended version of the JGAP library. Sampling the parameter space to form a new population in JGAP is guided by the constraints, dependencies and other details entered by



Select all the processors for your optimization subworkflow
☑ pepXMLComposer
☑ PeptideProphet
□ mzxmlDecomposer
☑ tandem

Evolution Steps: 40
Generation Size: 20
No change within x Generations: 3
Maximum runtime in minutes 1440
● Local Execution
○ Grid Execution

tandem
PeptideProphet
pepXMLComposer

□ FASTA_File
☑ MME_plus
☑ MME_minus

MME_plus Type: double
Min value 0.0
Max value 25
Function Type NONE
Function value 0.0
● round to #.##

(MME_plus - MME_minus) ☑ abs > 0.2

Figure 5.5: The screenshot shows the MS-based proteomics workflow in the new optimization perspective implemented by the Taverna optimization framework. (1) workflow diagram, (2) general selection mechanism for the sub-workflow, (3) specific input window for parameter optimization sampling, (4) the window for specifying parameter dependencies (optional). The workflow shows the proteomics experiment to be optimized, the greyed out components are not re-executed during the optimization.

the user. The resulting parameter set samples are returned to the plugin. The optimization plugin initializes the corresponding set of Taverna sub-workflows (cf. Figure 5.4(3a, 3b, 3c)). Then the optimization framework manages the parallel execution of the instantiated sub-workflows and forwards the results back to the parameter plugin after execution. The optimization parameter plugin extracts the fitness values and forwards them to the JGAP library, which evaluates the relative parameter set performance (cf. Figure 5.4(7a, 7b)). Afterwards, a new generation is created and the evolution continues until the optimal result is found or a termination criterion is reached." (Modified from [Holl2013f].)

This final result is presented to the user, who can store it as a file including statistics and meta-data of the optimization process or as a Taverna conform workflow input file for sharing purposes.

5.2.2 Discussion

A plugin for the optimization of scientific workflow parameter was described in this section. It plugs into the generic optimization framework described in Chapter 4 by implementing the required interface methods. The procedure of building this plugin is a first reference for further implementations of optimization plugins, as the interface and required methods remain the same. Therefore the plugin architecture and implementation can be used as a template for higher level and other optimization algorithms.

Due to the generic framework, developers do not have to deal with Taverna internals such as parallel execution, security mechanisms, data handling or GUI implementation, as described in Chapter 4. This allows keeping the implementation simple and slim, focused on the general optimization problem at hand.

The target workflow does not need to be specially adapted for the optimization process, except that parameters must be available as input parameter in Taverna and not as a constant value. Only a sub-workflow needs to be selected for optimization.

The plugin uses JGAP, a Genetic Algorithm library to perform the optimization. This library is hard connected to the plugin. Due to further extension, it would be possible to decouple the plugin and JGAP, nevertheless, the set-up of an optimization process is specific to the used library and could be tightly coupled.

"A crucial issue especially for non-experts in optimization is the parameterization of the optimization algorithms themselves. Currently, the optimization set-up includes crossover rate, mutation rate, and number of generations and population size of the Genetic Algorithm, which may still need to be adapted to the search space. Initializing these values could be supported in the future by automated tools [Ochoa2012] or based on provenance, which will be outlined in Chapter 6." (Modified from [Holl2013f].)

5.3 Evaluation of the Parameter Optimization Plugin

In order to evaluate the advantage of the developed parallel execution mechanism, the generic optimization framework and the example parameter optimization plugin, several real-world use cases from the life science domain will be experimentally tested in this section. The focus is on parameter optimization, whereas the motivation and scientific output differs for the individual use cases. To begin with, different configurations of the Genetic Algorithm were tested to i) find a reasonable minimum execution environment and ii) find good values for the crossover and mutation rate. These test runs are especially made for scientific workflow optimization and may not be of any general impact. For general benchmarks, the Rosenbrock Function [Rosenbrock1960] and Goldstein & Price Function [Goldstein1971] have been tested. In general, a mutation rate of 0.1 and crossover rate of 0.8 performed the best, especially for a small number of executions (other values may perform better for other specific optimization problems). In the following four sub-sections, four different use cases are presented.

5.3.1 Proteomics Workflows

As described in Chapter 3.3, proteomics aims at the fundamental analysis of proteins to gain details about their structures and functions. One goal is to determine the composition of a sample by measuring the masses of the molecules. The use case at hand optimizes the protein identification via tandem mass spectrometry and was elaborated in cooperation with Magnus Palmblad and Yassene Mohammed from the Leiden University Medical Center. The results of this investigation have been prepared and submitted for publication in [Holl2013e].

Motivation: "Peptide-spectrum matching is one of the most ubiquitous components of data processing workflows in mass spectrometry based proteomics. A large number of algorithms, embedded in free or commercial software packages, have been developed to address this task. However, the use of these algorithms invariably requires selection of data-dependent parameters, such as the Mass Measurement Error (MME) tolerance. MME values have a large influence on the total number of identified spectra. The optimal choice of these parameters depends on the species, sample preparation, enzymatic digestion, chromatography, mass spectrometer, data acquisition settings and database search algorithm. In inter-laboratory comparisons where the same data were made available to many different research groups, the number of returned identified peptides varied by almost an order of magnitude [PRG2013], demonstrating that finding good, or optimal, parameters for the peptide identification algorithms is a non-trivial problem.

Well-selected algorithms and parameters require significant knowledge about the data as well as the algorithms themselves. Typically, the choice of algorithm and parameters is determined by the users' experience and expert knowledge about the experiment, instrumentation and data quality. Hence, researchers tend to use default or estimated optimal parameters." (Modified from [Holl2013e].)

Workflow: A simplified version of the workflow used for optimization is shown in Figure 5.5. It matches fragment weights obtained by mass spectrometry to a database for identification of peptides and evaluates these findings.

The database matching process is conducted by *tandem*, which is executed on the distributed computing infrastructure as the application X!Tandem [Craig2004] from the TPP toolbox [Keller2005]. Tandem uses an mzXML file as input (file containing the weights of the measured spectra) and a database file in fasta format (FASTA_File). The mass spectra are only compared against those theoretical peptides whose mass is within a user-defined maximum Mass Measurement Error (MME+,MME-) from the experimentally determined mass of the whole peptide (the precursor). These parameters, MME+ (mass measurement error plus) and MME- (mass measurement error minus), are also tunable in the workflow.

After the identification of spectra, PeptideProphet [Keller2002] is used to estimate the probability of each assigned peptide-spectrum match by a mixture model of the X!Tandem score distribution, assuming there will be some correctly and some incorrectly identified spectra. Due to performance reasons, tandem was executed in parallel as described in Chapter 3.3. The mzXML_Composer was not included in the optimization process, as the splitting of files remains the same and is not needed for each iteration. The nrOfDaughters defines the number of data splits. The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3693.html.

Fitness: PeptideProphet concludes with a value for the estimated number of correctly identified spectra from doubly charged precursor. This was divided by the total number of tandem mass spectra to serve as a fitness value (maximization) for this use case. This value is referenced in the following as *2_num_corr*.

Data input: "As a prokaryote with a small genome and limited number of modified peptides, an *Escherichia coli* (*E. coli*) whole-cell lysate, prepared as described by Mostovenko et al. [Mostovenko2011] was used. This sample was analyzed both by ultrahigh-resolution TOF (time-of-flight) (named *E. coli*) and by an ion trap (named hybrid *E. coli*). As a eukaryote with a larger genome and frequently modified peptides, a sample from human dendritic cells was analyzed on the ion trap (named human). UniProt reference proteome data for *E. coli* (April 2013, 4,439 sequences and same number of decoys) and Homo sapiens (April 2013, 89,601 sequences including isoforms and the same number of decoys) were used for peptide identification." (Modified from [Holl2013e].)

Optimization:

Optimized parameters: MME+, MME-

Fixed parameters: mzXML_File, FASTA_File, nrOfDaughters

User constraints for parameters: $MME+ \in [0, 0.5]$ (double), $MME- \in [0, 0.5]$ (double)

User set dependencies: $|MME + -MME - | < 0.2$

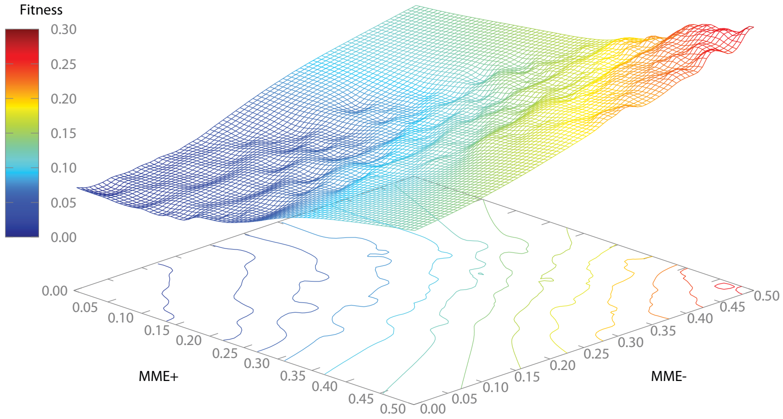


Figure 5.6: Plot of all fitness values obtained in the first iteration of the optimization of the Proteomics workflow. The surface was interpolated and visualized outside Taverna using gnuplot with the dgrid3d and contour base functions.

1st iteration:

Used data set: *E. coli*

Default values for parameters: $MME+ = 0.35$, $MME- = 0.35$

Fitness for default values: $2\_num\_corr = 0.1596$

Expected optima MME values: between 0.3 and 0.4

Results:

Optimization of the *E. coli* dataset obtained a fitness value of 0.2636 with $MME+ = 0.486$ and $MME- = 0.4672$, which is already close to the limits of MME+ and MME-.

However, the expected optimum value was not found. Due to this unsatisfactorily result, an investigation of all explored solutions was undertaken and illustrated in Figure 5.6. This result curve suggested that the global optimum is beyond the expected limits of the search space. Therefore, a second optimization was initiated with enlarged limits for MME+ and MME-.

2nd iteration:

User constraints for parameters: $MME+ \in [0, 25]$ (double), $MME- \in [0, 25]$ (double)

Used data sets: *E. coli*, hybrid *E. coli*, human

Default values for parameters: $MME+ = 0.5$, $MME- = 0.5$
Fitness for default values: *E. coli*: $2\_num\_corr = 0.2841$; hybrid *E. coli*: $2\_num\_corr = 0.2274$; human: $2\_num\_corr = 0.1019$

Results:
The results of the optimization are shown in Table 5.1 and for the hybrid *E. coli* also in Figure 5.7.

	E. coli	hybrid *E. coli*	human
Fitness (*2_num_corr*)	0.3134	0.2665	0.1148
MME+	7.32 Da	17.62 Da	7.95 Da
MME-	0.31 Da	5.57 Da	0.5 Da

Table 5.1: Results of the optimization of X!Tandem.

Scientific use:
The optimum MME- and MME+ windows were found to be asymmetric with a larger tolerance of positive MMEs. To the best of one's knowledge this have not been reported in the literature before. The phenomenon is explained in the following:

"In one dataset, the optimal positive MME was found along the ridge of +17.62 Da. This clearly visible ridge between 16 and 21 Da MME+ corresponds to the pyroglutamate/ammonia loss resonance. Including this MME range adds 437 peptide-spectrum matches (4% of all Peptide Spectrum Matchs (PSMs) with 1% False Discovery Rate (FDR)), which nearly all have assigned actual NH_3-loss *b*-ions as regular *b*-ions with 17 Da MME (see Figure 5.7 and sequence logo in Appendix A.1). The other optimal MMEs were found either just outside the default mass measurement errors (-0.31 or -0.50 Da) or just outside the MME corresponding to the precursor isolation window (-5.57, +7.32 or +7.95 Da) as illustrated in Appendix A.2.

Similar observations were recently and independently reported by three different groups at an international conference [Gibson2013; Wilmarth2013; Kim2013], including data from a Q Exactive Orbitrap [Michalski2011].

Parameter optimization lead to new insight into the data and the algorithms themselves – for example the ammonia loss *b*-ion spectra identified as peptides with N-terminal pyroglutamate. This phenomenon was not selected for investigation, but uncovered during the parameter optimization when allowing the parameters to vary over a wide range." (Modified from [Holl2013e].)

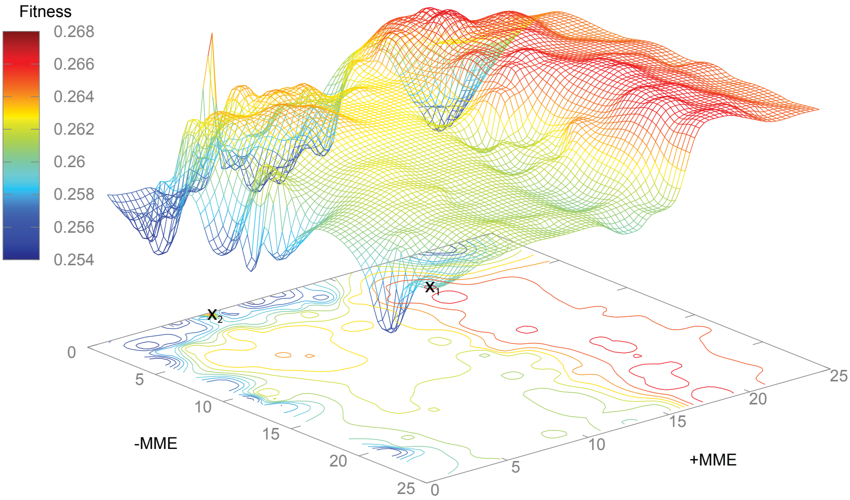


Figure 5.7: "Optimization of MME window for X!Tandem on the hybrid *E. coli* dataset (optimum marked with x_1), with fitness defined at the number of correctly identified spectra from doubly charged precursor divided by the total number of tandem mass spectra. The ridge corresponds to the pyroglutamate/NH_3-loss of the Peptide Spectrum Match. The surface was interpolated and visualized outside Taverna using gnuplot with the dgrid3d and countour base functions. " (Source: [Holl2013e].) x_2 shows the optima for interchanged values from the hybrid *E. coli* and human data set.

Optimization runtime: The population size was set to 20. As abort criteria, no change within 3 generations was selected and a maximum runtime of 24 hours was defined. For the execution, 4 CPUs per node were requested. The *E. coli* optimization terminated after 6 generations with a runtime of 4 hours. One single workflow execution for the *E. coli* dataset had a runtime between 7 and 12 minutes (depended on the selected MME+ and MME- value). The hybrid *E. coli* data set terminated after 5 generations and a runtime of 4:30 hours. The human data set runtime was 10 hours and terminated after 6 generations.

Further investigations:

To assess the dependency of data and the optimum value, the results of MME+ and MME- have been interchanged for the found optima. The result can be seen in Table 5.2.

These results let assume that the found optima are unique for the specific data. Nonethe-

	MME *E. coli*	MME hybrid	MME human
E. coli	**0.31314**	0.3093	0.3122
hybrid	0.2556	**0.2665**	0.2568
human	0.1143	0.1136	**0.1148**

Table 5.2: Fitness values of interchanged MME+ and MME- values.

less, the found optima are close together and especially the hybrid *E. coli* data set seems to have another local optima around the ridge of the *E. coli* and human data set optima (cf. x_2 in Figure 5.7).

Extended Parameter Optimization for X!Tandem

The presented workflow (cf. Figure 5.5) was also optimized including two additional parameters. Therefore, two input parameters were added to the Proteomics workflow presented in Section 5.3.1, namely the number of missed cleavages and a boolean to allow the enzymatic specificity (fidelity) to be 'full' (default) or 'semi-tryptic'. For these parameters, user constraints were set: $cleavage \in= [0,4]$ (integer), $specificity \in \{full, \textit{semi-tryptic}\}$ (boolean).

During the digestion of a peptide, the enzymatic reagent (trypsin) may not cleave the peptides at all possible sites. Thus fragments occur, which may contain several missed cleavages. In order to integrate these into the search one can define the maximum number of allowed missed cleavage sites within a peptide (*cleavages*). Especially if a researcher does not know whether the tryptic digestion went well or not, a *larger* number would be selected. The enzymatic specificity defines the allowed type of these cleavages. 'Full' – strict tryptic cleavage – means that both ends of the peptides have to be tryptic. 'Semi' means that only one site of the cleavage is tryptic and the other may be of any residue ($specificity$). Both, but especially the second parameter, influence the runtime of the workflow, as more allowed cleavages enlarge the peptide sequence search space.

The tested data set was *E. coli* and the other parameter ranges were the same as in the above described use case. Using the default values (cf. Table 5.1): missed cleavages counted *2* and the fidelity was 'full'. The results are shown in Table 5.3. It becomes visible that $cleavage = 1$ in the average performs better than $cleavage = 2$, however improvements are less than 1% and negligible.

Scientific use: "The results show that a value of *1* for the number of missed cleavages produce the optimal results. This reflects that the digestion went very well, but occasional

	Fitness (2_num_corr)	MME+	MME-	# cleavage	fidelity
default	0.3134	7.32 Da	0.31 Da	2	full
optimized	0.3142	7.17 Da	0.28 Da	1	full
default/optimized	0.3143	7.32 Da	0.31 Da	1	full
optimized/default	0.3142	7.17 Da	0.28 Da	2	full

Table 5.3: Results of the optimization of X!Tandem including 4 different parameters.

cleavage sites were missed in some peptides. These can be minor products from highly abundant proteins, but including these in the search still increases the number of identified spectra (if not the number of unique peptides).

The optimum setting for enzyme fidelity was ’full’, which produces the smallest search space. This can be explained by the used highly specific proteomics grade of trypsin. However, looking at the peptide identifications produced used as ’semi’ fidelity, one immediately notices that most of the peptides with one non-tryptic cleavage derive from the protein N-terminus sans the N-terminal methionine. These peptides are in fact most likely not ’semi-tryptic’ in the enzyme fidelity sense, but reflect a prior event where the N-terminal methionine was removed by Methionine aminopeptidase (MAP) *in vivo*. A sequence analysis of these peptides (Appendix A.3) shows the known pattern for MAP-derived peptides [Waller1963; Frottin2006; Xiao2010], suggesting that most of the identifications are correct.

This is therefore another example where expanding the search window (by allowing semi-tryptic fidelity) leads to the discovery of something biologically meaningful within the data but not actually related to the fidelity of trypsin." [Palmblad2013]

Optimization runtime: The population size was set to 20 and the termination criterion was set to maximum 40 generations, within 24 hours and no change within 3 generations was defined. The optimization process terminated after 10 generations and took 21 hours. The runtime of one workflow execution was between 11 minutes and 1 hour depending on the input parameters. Especially the allowance of the enzymatic specificity to be ’semi-tryptic’ increases the runtime as the spectra are being searched against many more theoretical peptides, as only one of the ends is required to be tryptic (C-terminally of lysine and arginine in the protein sequence).

5.3.2 Ecological Niche Modeling Workflows

Ecological Niche Modeling (ENM) has become a widely used approach to analyze species distributions and to predict changes in biodiversity patterns [Wiley2003; Guinan2009; Kulhanek2011]. "The idea of niche modeling is based on G.E. Hutchinson's definition of the realized niche, where a set of environmental factors or a multidimensional space of resources (e.g. light and structure), determine the persistence of a species [Hutchinson1957]. Thus, with relatively few variables characterizing the abiotic environment of the species in the form of geo-referenced raster layers, potential distribution models can be generated (cf. Figure 5.8). Biotic factors such as predation and/or competition also determine the niche of species, but in marine ecosystems abiotic factors (e.g. salinity and temperature) are considered as the major features limiting the distribution range of many species [Paavola2005; Guinan2009].

Motivation: In order to obtain valid models for species niches, it is important to specify the appropriate set of parameter values for a given input data set (occurrences, mask, and

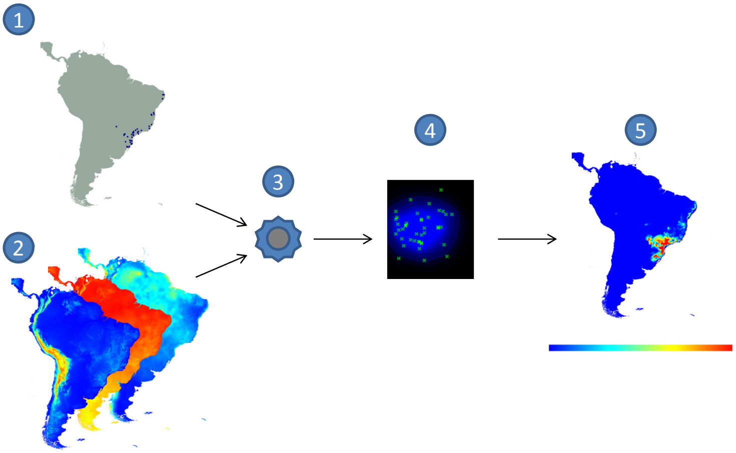


Figure 5.8: "General principle of Ecological Niche modeling. (1) Species occurrence points. (2) Set of spatially explicit environmental variables that influence the distribution of the species. (3) Ecological niche modeling algorithm. (4) Generated model in the environmental space. (5) Model projected back into the geographical space, showing habitat suitability for the species from blue (unsuitable) to red (suitable)." (Source: [openMod2013], [Holl2013c])

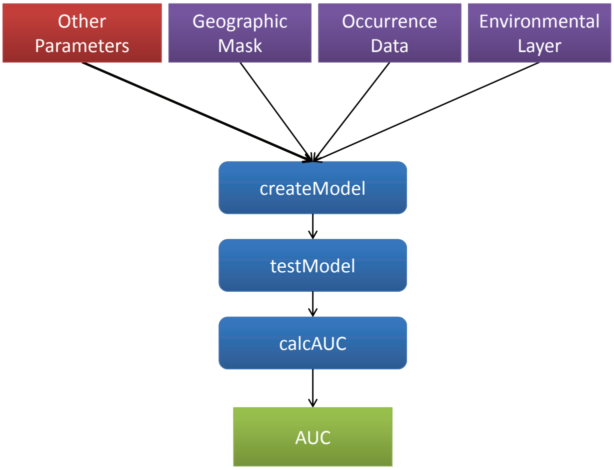


Figure 5.9: The abstract ENM workflow uses occurrence and environmental data to model species distribution based on a variety of algorithms, including Maxent, Support Vector Machines and others.

layers). The best parameter values can vary between different input data sets and hence are difficult to know beforehand." (Modified from [Holl2013c].) "The ENM workflow analyses rely on default parameter values or parameter free algorithms. In some cases resulting models are not sufficiently accurate." [Obst2013]

Workflow: The current ENM workflow [BioVel2012a] as shown in Figure 5.9 (Taverna workflows used for optimization, see Appendix A.4 and A.5) was developed by the BioVeL project [BioVel2012b; Vicario2012] and uses occurrence and environmental data to model species distribution based on a variety of algorithms, including SVM, Maxent, and others [Souza Muno2011]. The algorithms are remotely executable through openModeller [openMod2013] Web services. All algorithms take a mask, layers and geographic coordinates of a species as input and specific parameters to generate the potential distribution models. The mask describes the geo-spatial mask selection, a species specific geographic area for projection. The environmental layers describe the species specific conditions such as temperature, precipitation, or salinity, which are likely to affect the distribution. The number of pseudo absences is required by many of these algorithms and defines a number of random, environmentally or spatially generated points where the species does not occur.

Within the first step of the workflow, the input parameters are prepared and the *model creation* operation is called on 90% of the input points. The model then represents the suitable conditions of abiotic for a given species. After creating the model, the *test model* operation is called. This operation tests the model, by using the 10% points left out of model creation. The test operation calculates the Receiver-Operating Characteristic (ROC) curve, and the Area Under the Curve (AUC).

The SVM-based Web service uses a Support Vector Machine to build the prediction model. Maxent-based Web service uses a Maximum entropy modeling technique to find the distribution of maximum entropy. The algorithms build a model, which represents the possible combination of biotic and abiotic factors – the niche for a specific species. This model predicts the current distribution of the species but it can also be designed taking climate forecasts into account. These models predict the species distributions under specific forecasts in the future. The runtime of the workflow is for SVM between 8 minutes and 10 minutes and for Maxent between 10 minutes and 13 minutes.

Fitness: "The model is built using a 10-fold cross validation. For each of these model creations, a statistical evaluation of the accuracy of the model prediction is performed. For this, the AUC of the ROC curve for each species was estimated and used to test the model's discriminatory power. Finally, the workflow calculates the average AUC, which can be used as fitness value." [Giovanni2013].

Data Input: The data set of the Harmful Algae Blooms (HABs) (*Prorocentrum minimum* (Dinoflagellata)) contains 172 occurrence records of the species in the North East Atlantic, a geographic mask for the same region, and a set of environmental layers that drive the distribution of the species (mean sea surface temperature, mean salinity, mean photosynthetically available radiation). The data set of the Pacific oyster (*Crassostrea gigas*) contains 121 occurrence records in coastal European environments, a geographic mask for the same region, and a set of environmental layers that drive the distribution of the species (mean annual distance to land, maximum depth, mean annual surface salinity, mean annual surface temperature, mean annual sea ice concentration). Both data sets can be found at the Global Biodiversity Information Facility Web page [GBIF2009]. More details of these species can be found in Appendix A.2.1.

Optimization:
Used algorithms: SVM, Maxent
Optimized parameters: SVM: gamma, cost, numberOfPseudoAbsences;
Maxent: numberOfIterations, quadraticFeatures, productFeatures, hingeFeatures, thresh-

oldFeatures, autoFeatures, minSamplesForProductThreshold, minSamplesForQuadratic, minSamplesForHinge
Fixed parameters: SVM: probabilisticOutput, nu, coef0, degree, kernel_Type, svm_Type; Maxent: numberOfBackgroundPoints, terminateTolerance
User constraints for parameters: SVM: *gamma* $\in [0,10]$ (double), *cost* $\in [0,156]$ (integer), $numberOfPseudoAbsences \in [200,600]$(integer);
Maxent: $numberOfIterations \in [500,1000]$ *(integer)*, $quadraticFeatures, productFeatures, hingeFeatures, thresholdFeatures, autoFeatures \in [0,1]$ *(boolean)*, $minSamplesForProductThreshold \in [70,90]$ (integer), $minSamplesForQuadratic \in [5,15]$ (integer), $minSamplesForHinge \in [10,20]$ (integer)
User set dependencies: –

Results: Please refer to Table 5.4 and Table 5.5 for optimization results of SVM and Maxent workflows.

Scientific use: After the optimization of the model, the optimal identified parameters were used to project the model of each species today and for the *Crassostrea gigas* for the year 2050. The projections have been visualized and the results are shown in Appendix A.7, A.6, A.9, and A.8. All projections are more accurate than using the default parameters. This can be clearly seen by comparing both, the default and optimized projections of the SVM algorithm as shown in Appendix A.7 and A.9. Construing the optimized projections lead to the following scientific conclusions:

"The Harmful Algae Blooms (*Prorocentrum minimum*) are not likely to establish algal blooms in the central and Eastern Baltic. The main risk zone for Prorocentrum HABs are in the area of Kattegat, Skagerrak, and the Western Baltic.

	default	optimization	default	optimization
gamma	0	2.36	0	0.38
cost	1	9	1	5
numberOfPseudoAbsences	0	363	0	299
AUC	0.8794	0.9207	0.9486	0.9690
species	*Prorocentrum minimum*		*Crassostrea gigas*	

Table 5.4: Results for default values and optimization of SVM algorithm.

	default	optimization	default	optimization
minSamplesForHinge	15	17	15	13
hingeFeatures	1	1	1	1
thresholdFeatures	1	1	1	0
numberOfIterations	500	930	500	625
productFeatures	1	1	1	1
autoFeatures	1	1	1	1
minSamples-ForQuadratic	10	6	10	15
quadraticFeatures	1	0	1	0
minSamplesFor-ProductThreshold	80	73	80	63
AUC	0.9243	0.9392	0.9522	0.9657
species	*Prorocentrum minimum*		*Crassostrea gigas*	

Table 5.5: Results for default values and optimization of Maxent algorithm.

The habitat of the Pacific oyster (*Crassostrea gigas*) will migrate northward in the coming decades. Typical oyster banks in the south (Brittany, Normandy) will become unsuitable, while northern locations (West coast of Norway and Northern Ireland and Britain) will gain suitability for this species. This has important implications for aquaculture industry. Some of the optimized models give more accurate projections of species distributions. Without the optimization, ENM workflow analyses rely on default parameter values or parameter free algorithms such as the Mahalonobis algorithm. In both cases, resulting models may be not sufficiently accurate." [Obst2013].

Optimization runtime: The Web services used in this workflow are executed on a Web server situated in Brazil. This Web server is able to execute 10 requests in parallel. This entailed an non-responding Web server if the population size was set to larger than 16. Additionally, the server is not available for other users during the optimization process and an alternative server was not available. Due to this, strict time restrictions were set to the optimization runs (maximum 12 hours). In this example, the results are the best for the given time and might not represent the optimal parameter sets. The SVM-based workflow was executed around 112 times and the Maxent-based workflow around 96 times.

5.3.3 Biomarker Identification Workflows

An important topic in disease diagnosis and biomedical applications is the identification of potential biomarkers, which can for example be represented by differentially expressed genes in microarray data under a specific disease condition. Most traditional statistical methods can discriminate between normal and differentially expressed genes, but their prediction performance across datasets are poor [Chen2011]. Thus, there is a strong tendency to robustly identify differentially expressed genes.

The selection of relevant genes from large high-dimensional gene expression data can be applied by feature selection methods. Two feature selection methods have been implemented here are based on machine learning techniques developed by [Abeel2010]: RFE and EFS. The Recursive Feature Elimination algorithm (RFE) is a recursive process, which uses the absolute values of the weights of each dimension in the hyperplane of a Support Vector Machine (SVM) as importance of each feature (gene) in the dataset. Ensemble Feature Selection (EFS) is based on an ensemble concept, where multiple feature selections are combined to increase the robustness of the results. To get a robust and valuable evaluation result, models are built by sub-sampling the original dataset several times. The result represents a ranked feature list from which highly ranked genes are selected as potential biomarkers. These potential biomarkers may be tested in a further step in the laboratory. The biomarker identification workflow was developed in conjunction with the Department of Bioinformatics of the Fraunhofer Institute for Algorithms and Scientific Computing.

Motivation: The main concern of the classification hyperplanes of SVMs in the field of biomarker identification is its interpretability and robustness for prediction of potential biomarkers. It is important to have properly calibrated machine learning methods available, which identify relevant genes and include these in the upper list of ranked biomarkers. Therefore, there is a need to identify of a set of classifier features to attain maximum predictive power.

Workflow: Figure 5.10 illustrates the workflow for biomarker identification. The first two components split the original data set into several sub-sampling sets. The RFE component performs the machine learning approach by executing several instances of a SVM, each of which consuming one sub-sample data set. The RFE uses the weight of

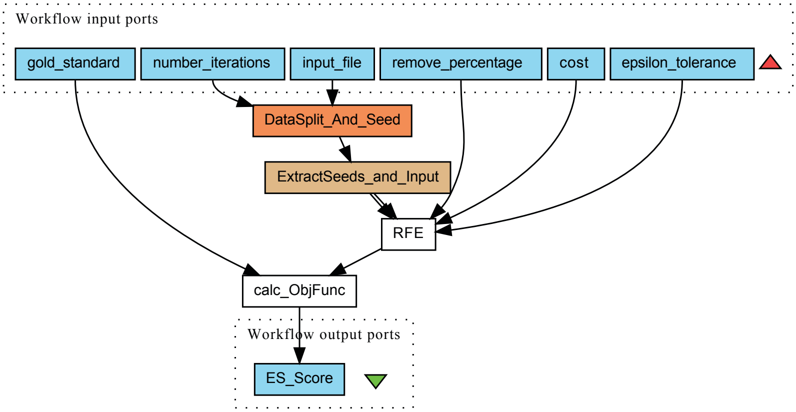


Figure 5.10: The biomarker identification workflow. The RFE can be substituted by an ensemble feature selection (EFS). The fixed input parameters of the RFE/EFS are not shown.

each feature to i) rank the features from most important to least important and ii) removed iteratively the least important features. The execution of the SVM was performed on the Grid, as typically around 100 parallel instances of SVMs are executed in one workflow run. The feature selection can also be substituted by an Ensemble Feature Selection (EFS). The EFS adds an additional level of sampling to the RFE, namely bootstrapping. The fitness value is calculated by calc_ObjFunc and named *ES_Score*. The workflows are available at myExperiment: http://www.myexperiment.org/workflows/3694.html and http://www.myexperiment.org/workflows/3695.html.

Fitness: The calculation of the objective function is performed by comparing the ranked gene list against a so-called '*gold standard*'. The gold standard was produced using a *t-test* as described in [Smyth2005] and lists the top 40 ranked genes. The fitness measure was calculated by comparing this list with the signature of the feature selection by the RFE utilizing an F-measure.

Data Input: The data input set of the mouse was taken from the publicly available GEO database [Edgar2002]. The used Affymetrix GeneChip microarray data of Huntington disease at 24 months of age (GEO accession GSE18551; [Becanovic2010]) contained 39637 features.

Optimization:
Optimized parameters: remove_percentage, cost, epsilon_tolerance;

EFS additional parameter: bootstrapping
Fixed parameters: input_file, number_iterations, gold_standard, svm_type, kernel_type, degree, gamma, coef0, nu, epsilon_loss, cache_size, shrinking, probability
User constraints for parameters: $remove\_percentage \in [0.01, 0.33]$ *(double)*, $cost \in [0, 256]$ *(integer)*, $epsilon\_tolerance \in [0, 2]$ *(double)*, $bootstrapping \in [1, 50]$ *(integer)*
Used data set: 24 month mouse Affymetrix
Default values for parameters: $remove\_percentage = 0.2$, $cost = 1$, $epsilon\_tolerance = 0.001$, $bootstrapping = 40$
Fitness for default values: RFE: fitness = 0.3076; EFS: fitness = 0.3846

gene	gold standard	default	optimized	default	optimized
Ddit4l*	2	1	1	2	2
Wt1*	3	27	14	27	7
Dleu7	4	85	25	43	20
Slc45a3*	5	19	6	9	5
Gsg1l*	13	244	138	115	29
Entpd7	17	308	163	183	75
Spata5	20	69	18	38	28
Grasp	25	617	383	362	154
Dmpk	38	611	221	424	147
fitness		0.3076	0.4871	0.3846	0.5641
algorithm		*RFE*		*EFS*	

Table 5.6: Comparison of the optimization results. The genes marked with * have been tested in the laboratory.

Results: The optimized RFE fitness was 0.4871 with $remove\_percentage = 0.11$, $epsilon\_tolerance = 0.5498$, and $cost = 237$. The optimized EFS fitness was 0.5641 with $remove\_percentage = 0.28$, $epsilon\_tolerance = 1.9762$, $cost = 69$, and $bootstrapping = 33$. The optimization results are shown in Table 5.6.

Scientific use: Table 5.6 shows the results obtained from the biomarker identification workflow optimization. "The original analysis by the authors [Becanovic2010], evaluated the top 10 ranked genes obtained by their analysis which are differentially expressed in 24 months mouse affected by Huntington's disease. These 10 genes were experimentally tested in the laboratory and the four genes marked with a star were found to be differentially expressed in the Huntington disease. As it was not possible to reproduce the authors results using the same statistical analysis (*t-test*), an own implemented t-test (top 40) was used as '*gold standard*'. In the top 40 of this t-test result, 9 genes out of the authors top 10 list were present, listed in Table 5.6. The optimized RFE was able to rank all genes higher than using the default parameters. It identified 5 out of 10 instead of default 3 within the top 40 gold standard genes. Similarly, the optimized EFS identified 6 out of 10 instead of default 4 within the top 40 gold standard genes. In addition, EFS was able to identify 3 out of 4 validated genes in top 10. The EFS method was expected to perform better than RFE as it applies an additional level of sub-sampling, namely a bootstrapping method. Due to this supplementary level, average weightings are calculated and the final result is more robust and stable. This improvement motivates to optimize additional biomarker identification workflows used with other species, diseases or platform formats." [Bagewadi2013]

Optimization runtime: The population size was set to 20 and the termination criterion was set to maximum 40 generations, within 24 hours and no change within 5 generations was defined. The optimization process terminated after 16 generations and took 3:20 hours. The runtime of one workflow execution was between 3.8 minutes and 5.2 minutes depending on the input parameters.

5.3.4 Protein Structure Similarity Workflows

"A recurring task in data driven biosciences is the development of predictors. Given a suitable set of training data, supervised prediction algorithms such as Support Vector Machines are trained, and the resulting model can then be used to classify new input data or to predict the value of a particular property. As an example, a training of a local protein structure predictor was selected. This predictor estimates the structural similarity of a protein segment for which only the sequence profile is known. The local backbone structure of the protein as a vector of 8 consecutive dihedral angles was defined. By using a training set that contains the sequence profiles as vector features and the structure similarity to a particular reference structure, one can train a model that estimates the similarity of a protein sequence and the reference fragment in terms of the root mean square deviation. Support Vector Regression (SVR) [Smola2004], a technique closely related to SVM, can be used to train such an estimator. A set of such estimators can for instance be used for the prediction of the secondary structure of proteins, an important technique in structural bioinformatics." (Modified from [Holl2012b].)

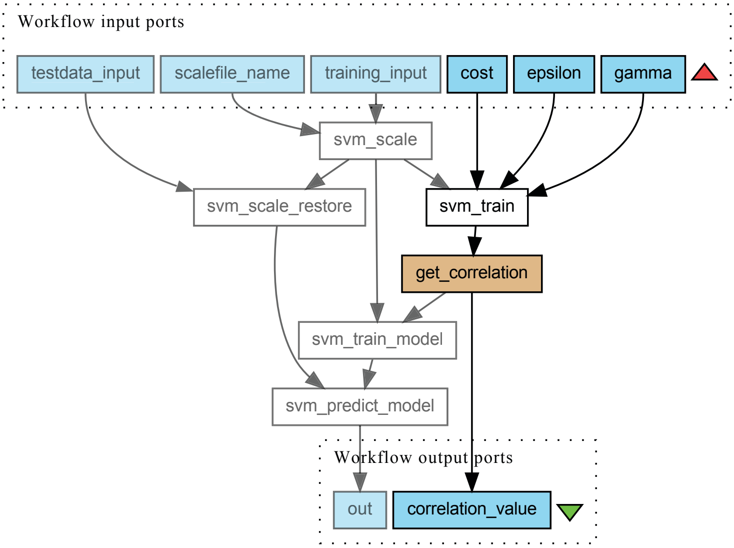


Figure 5.11: The screenshot shows the support vector regression workflow in Taverna. The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3696.html.

Motivation: The SVR training was performed by using a so-called Radial Basis Function (RBF) kernel. This kernel function has one parameter *gamma* that specifies the width of the Gauss function of the RBF. The function for penalizing errors has a parameter *epsilon*, to specify the width of an insensitive zone and there is a regularization parameter *cost* that penalizes the complexity of the model. Optimizing these three real-valued parameters by a manually guided parameter sweep, 420 trials have been used.

Workflow: In Figure 5.11 the set-up of the regression training as a Taverna workflow is shown. The server components in this case are the scaling and training programs from LIBSVM, a SVM library [Chang2011].

Fitness: The squared correlation coefficient, that is one of the outputs of the LIBSVM program and has its optimum at 1, was used as fitness function.

Data Input: The SVR training set is comprised of a sample of 5000 dense vectors representing PSI-BLAST profiles of 15-residue sequence windows obtained from proteins with known structure. The label of each vector is the dihedral Root Mean Square Deviation (RMSD) between the central 5 residues of the sequence window and a reference, in this case as extended 5-residue peptide structure as found in beta strands. The reference dihedral angles have been taken from the protein block ’d’ of the structural alphabet developed by Alexandre de Brevern [De Brevern2000].

Optimization:

Optimized parameters: cost, gamma, epsilon

Fixed parameters: test_input, scalefile_name, training_input

User constraints for parameters: $cost \in [2^0, 2^8]$ (integer, exponential function), $gamma \in [2^{-12}, 2^0]$ (double, exponential function), $epsilon \in [0.0, 10.0]$ (double)

Optimal values for parameters from parameter sweep: $cost = 64$, $gamma = 0.03125$, $epsilon = 0.75$

Fitness after 420 parameter sweep executions: 0.527902

Result: The optimized fitness was 0.588481 with $cost = 61$, $gamma = 0.03342$, $epsilon$ $= 1.5758$ with using around 100 workflow executions.

Scientific use: "The fully automated optimization using GA found an optimum on par with a parameter sweep technique that required four times as many samples, thus being more time and resource efficient. As the real use case would optimize a set of several hundred such estimators, each with a different set of optimal parameters, automated optimization would become a requirement." [Holl2012b]

Optimization runtime: This example aimed at finding a better correlation coefficient than the parameter sweep within fewer workflow runs. Whereas more than half of the workflow runs had a better fitness value than the parameter sweep, 114 workflow executions have been required to get a result 10% better than the parameter sweep. However, this is around four times less than required for the parameter sweep. The runtime of the optimization was 2:20 hours whereas one workflow execution took 1.4 minutes.

5.3.5 Discussion

In the last sub-section, four different case studies, illustrating real-world optimization problems, have been experimentally tested. Within all four use cases, better solutions than the default values or parameter sweep results have been obtained. The first use case showed the usability of the abort criteria for *maximum time* and *no change within a specific number of generations*. Performing this example brought new insides on database search software in the proteomics area. With the extension of two to four parameters the scalability and robustness of the system was shown. This parameter adding produced a four dimensional search space that potentially requires an 8-times longer search, whereas the real search time only required twice as many generations to find an optimal solution.

The ENM use case illustrated usability while applying the SVM algorithm, with respect to a lower population size and hard time constraints. Due to the capacity of the server, only 16 instances have been performed in parallel and only a few hours have been available on the server for optimization. Nevertheless, the results lead to improvements, which are now all usable. Providing more computing power for instance in terms of a second server instance or several Cloud instances, more samples could be tested during the optimization and the overall result finally improved. This would have been required in order to optimize the Maxent algorithm properly. This algorithm has many parameters and the optimization only gained small improvements, which is based on the small number of performed executions. The usage of a dependency setting could also have helped. As this dependency description would be much more complex than a regular arithmetic expression, it could not be defined with the recently implemented dependency description mechanism. Instead, a more complex scripting-based mechanism would be required to cover such cases.

The last use case showed a common life science scenario: the optimization via parameter sweeps. Parameter sweeps have been the method of choice to test different parameter set-ups and obtain an appropriate solution. The comparison of a parameter sweep search and an heuristic optimization might be redundant, however it could be shown that a 10% better result was achieved by using four times less samples.

Regarding the already available termination constraints, especially in the area of scientific workflows, additional criteria may be taken into account. At the moment, the user can limit the time spend for the optimization. Sometimes, one may not only want to restrict the overall used time, as queuing on a machine, upload and download rates are included in such a measure. To cover other conditions, for example a limit of used CPU hours would be required.

Regarding other recent design decisions, the implemented parameter plugin could be improved. One aspect is the handling of constraints. Recently, all values above or below the limits are just being ignored. Another commonly used method would be to add a penalty to those values. Additionally, many other bio-inspired methods [Coello2002] exist. In doing so, better solutions outside of the defined search space could be found. Especially in the proteomics use case illustrated in Section 5.3.1, such a method would have helped to identify the potential of a larger search space much faster. However, enlarging the search space to 25 Da caused a three times longer execution time. Thus, the calculations themselves may be without use for the optimization process, moreover an enlarged search space might cause much longer execution times. Due to already existing compute time problems, a penalty function for constraints was not identified to be useful within these thesis examples.

The Genetic Algorithm uses a real-coded parameter set-up, which is especially useful when performing parameter estimation for real-world problems [Wright1991; Deb2002a]. As the original genetic operators have been developed for binary-encoding, an adapted crossover and mutation operator are useful. Several different approaches have been reported by Herrera et al. [Herrera1998] or Wright [Wright1991]. These approaches mainly differ from the traditional operators in a way that they mimic the typical binary crossover and mutation operations. For example, the mutation is not performed by switching the gene value, moreover it is performed by multiplication of a random value. This thesis has used the BLX crossover operator but others could also be applied. The selection of an appropriate operator is very difficult, as each variation has a specific benefit under certain circumstances. However, this is a general problem of heuristic search methods and will be further discussed in Chapter 6.

5.4 Simulation of Workflow Structure Optimization

In Chapter 4.1.2, three different optimization levels have been presented. In the last section, parameter optimization was experimentally tested. Although the example plugin implemented within this thesis targets parameter optimization, higher level workflow optimizations such as component and topology optimizations can be experimentally simulated with this plugin. In this section, one of the introduced use cases is extended and evaluated regarding component optimization. A new use case is shown to simulate topology optimization

5.4.1 The Component Level

As already illustrated in Chapter 4.1.2, components used in a scientific workflow may implement the same algorithm or function but in a different fashion. Scientists want to evaluate which implementation performs best for a specific problem. This section shows results of the optimization of workflow components by reusing the implemented parameter optimization plugin. Therefore, each occurring parameter of each application is encoded as a single parameter in the Genetic Algorithm. Additionally, a flag is encoded as parameter in the GA to enable only one application for a specific set of parameter. In the following, the proteomics workflow, see Section 5.3.1, was further extended and optimized by component optimization.

Proteomics Workflow:
The use case proposed in Chapter 3.3 and optimized in Chapter 5.3.1 was extended to adapt component optimization. The original workflow finished with a list of peptides that could be tentatively matched for identification. This list still includes false positives and to identify or remove these, additional information can be incorporated. One way to do this is to train a so-called retention time predictor. The retention time can be extracted from the tandem mass spectrometry process and characterizes the time, a particular peptide needs to pass the chromagraph under certain conditions – between the injection and detection. This characteristic can be used to remove peptides that do not fit the predicted chromatographic behavior from the list of peptide matches [Palmblad2002].

The results of this investigation have been prepared and submitted in [Holl2013e]. *Motivation*: "There are a number of algorithms to predict peptide retention times, for instance one developed by Palmblad et al. [Palmblad2002] or two of which are already included in the RTCalc utility in TPP [Keller2005]: one based on SSRCalc [Krokhin2006] and one based on the artificial neural network (ANN) method by Petritis and co-workers [Petritis2006]. To demonstrate how a scientific workflow can select an optimal path for proteomics data analysis, a workflow to balance the quality (FDR or minimum Peptide-Prophet probability) of the training set and the prediction model was designed. It should find the model that can best predict the retention times of peptides within the same dataset and a specific quality measure. The rationale is that the simpler retention time predictors have fewer free parameters and will be more robustly fit by smaller training sets than the potentially better but more complex models, requiring much more training data." (Modified from [Holl2013e].)

Workflow: The workflow is shown as a stand-alone workflow in Figure 5.12. During one execution, only one path of the workflow is executed, regarding the flag. In doing so, no useless runs are created. The *probabilities* parameter has as input a list of peptide probabilities for each identified peptide from PeptideProphet.

One sub-workflow executes the retention time predictor developed by Palmblad et al. [Palmblad2002] (named *rt*). The other two sub-workflows apply algorithms out of the TPP toolbox, namely an ANN-based algorithm (named *ANN*) and the standard RTCalc algorithm (named *RTCalc*). "As RTCalc has its own hard-coded internal quality checks that generates an error message it aborts rather than produce a poor or overfitted model.

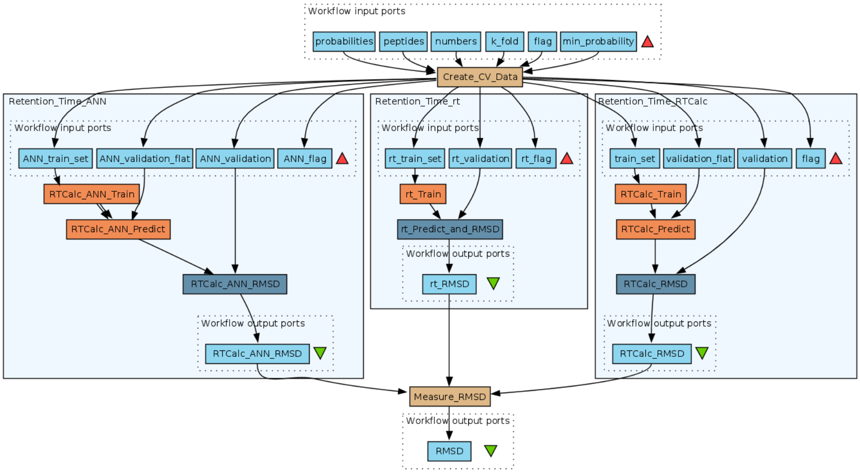


Figure 5.12: The workflow to perform retention time prediction with i) RTCalc (right hand sub-workflow), ii) rt (middle sub-workflow), or iii) ANN (left hand sub-workflow). The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3691.html.

These checks were disabled in the RTCalc source code to level the playing field and allow the optimization framework to independently find the right combination of parameters and algorithm.

Fitness: The quality of the retention time prediction was evaluated as a root mean square deviation (RMSD) for 10% of the peptides held back as a validation set, using the remaining 90% of the peptides to train the model. These 10% were then chosen at random 10 times so that each peptide was used exactly once for validation. " (Modified from [Holl2013e].) The RMSD was minimized during the optimization.

Data input: The optimized output file of PeptideProphet produced in Chapter 5.3.1 was used as input for this workflow.

Optimization:

Optimized parameters: min_probability, flag

Fixed parameters: peptides, probabilities, numbers

User constraints for parameters: $minimumprobability \in [0.5, 1]$ (double), $flag \in 1, 2, 3$ (integer)

Used data sets: hybrid *E. coli*

Result: RTCalc preformed best with a probability of 0.607, as shown in Figure 5.13.

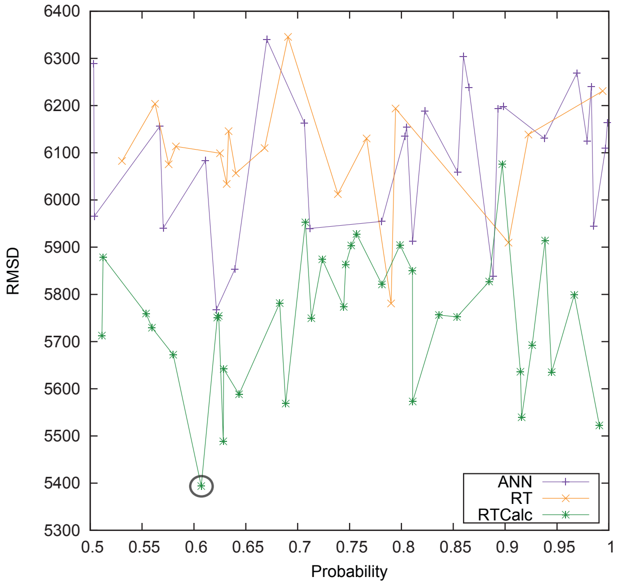


Figure 5.13: RMSD values of the optimization of i) RTCalc, ii) rt, and iii) ANN.

Scientific Use: The example use case shows that component optimization is useful when comparing the performance of different applications. The data formats remained the same and simplified the design of the example switch workflow.

"As expected, the number of peptides in the training sets used here were not sufficient to produce an accurate model using the artificial neural network algorithm. For more than 70-80 peptides in the training set, the RTCalc coefficient model seemed to perform best. For 52-70 peptides, the rt performed better, and for 21-52 peptides in the training set, only rt produced a model at all. No model was returned when having 21 peptides or fewer in the training set. In absence of quality checks, the minimum number of peptides required to produce a model is solely determined by the number of free parameters (terms) in the model. Figure 5.13 shows the retention time prediction accuracy as a function of model and minimum probability/training set size. " [Holl2013e]

5.4.2 Discussion

The last section illustrated a real-world use case for the optimization of workflow components. The result showed that the different applications achieve different performance measures regarding the probability value. This finding lead to new insights in data and algorithms themselves [Holl2013e].

Due to the lack of an ontology describing life science applications and their functions as well as input and output types, it is recently not possible to provide a fully automated optimization of workflow components. Thus, the developed parameter optimization plugin was used to find the optimal component. Therefore, a workflow was manually created to switch between the three different sub-workflows. Each sub-workflow was encoded by one flag, which implements switching the sub-workflow on or off (only one path is executed). This simulated set-up of component optimization would be challenging in practice due to the difficulty of building a switchable workflow. However, within another plugin targeting component optimization, this behavior could be similarly used and a switching workflow automatically created by the plugin.

During the optimization of components, parameter optimization has to be taken into account, likewise. Thus, component optimization does not notably differ from parameter optimization, besides the fact that the number of input parameter is much larger. One approach would be to constitute one gene on a chromosome as a flag to point out the evolved application. During the execution of one component, only the encoding block would be regarded for the specific run. Albeit this, the chromosome size would enlarge which would result in a larger required population size which would in turn enlarge the execution time. A serious pre-validator would be needed to predict reasonable exploration areas in the search space and neglect misguided solutions. Such pre-validators will be discussed in more detail in Chapter 6.

5.4.3 The Topology Level

As stated in Chapter 4.1.2, components used within a scientific workflow may also work in a switched order. This can be the case for example when using filtering or clustering mechanisms. Scientists want to evaluate, which workflow order performs best for a specific problem. This section shows an example use case utilizing two different filter mechanisms.

BLAST workflow:

A widely used experiment in the life sciences is to search a nucleotide or protein database for sequence similarities. BLAST is the common tool to perform such a search. Using the BLAST Web services, scientists can enter a target search sequence and BLAST matches this query sequence against a database. According to a calculated statistical significance the best matched sequences are returned.

Motivation: In order to reduce the size of the result set of sequences, scientists may apply different filter mechanisms. One such filter may select the top *k* sequences regarding the expected value (e-value), which can be used as a statistical significance threshold. Another filter may regard the maximum length of the matching sequences that orders the result set and filters the top *k* longest sequences.

Workflow: An abstract workflow of this experiment is shown in Figure 5.14. This workflow can be executed in the order $filter\_length \rightarrow filter\_e\_value$ or $filter\_e\_value \rightarrow filter\_length$. A scientist may want to know which path performs best regarding a specific data set. The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3706.html

Fitness: The fitness value was selected to be the relation of how many sequences of the original result and the final result occur in a gold standard after filtering (using a fixed number of top *k* sequences).

Data input: As example input, a protein family was used as gold standard and one exemplary sequence was applied for the search. The gold standard originated from the ELL-associated factor (EAF) family. The Pfam database entry can be retrieved from http://pfam.sanger.ac.uk/family/PF09816.

Optimization:
Optimized parameters: top_e_value, top_length, flag
Fixed parameters: database, sequence_id
User constraints for parameters: $top\_e\_value \in [30,70]$ (integer), $top\_length \in [30,70]$ (integer), $flag \in 1,2,3$ (integer)
Used data set: EAF family
Result: The maximum fitness was reached with the path $filter\_length \rightarrow filter\_e\_value$ using the top 67 sequences for length filter and the top 35 for the e-value filter. 34 sequences of this top 35 list occurred in the gold standard. On the contrary, only 31 sequences from the original top 35 list occurred in the gold standard.

Scientific Use: The example use case shows that topology optimization is useful especially when using filter or clustering mechanisms. The data formats remained the same and the example workflow could be designed without using data transformation components.

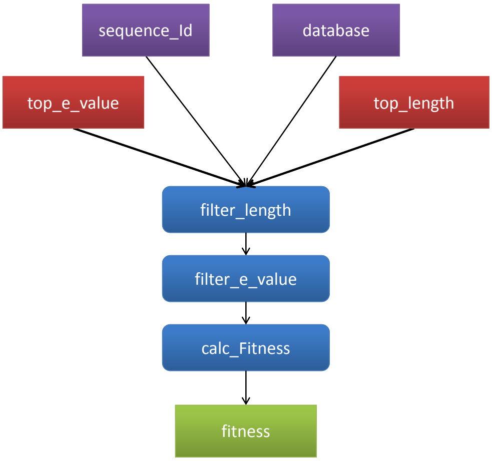


Figure 5.14: An abstract workflow to perform a BLAST search including two filters. The order of the filter may be changed and optimized.

5.4.4 Discussion

The exemplary use case shown in the last sub-section took the e-value and the length of a sequence match as a filter. From this, a workflow with two alternative paths was created. The optimization was simulated by the parameter optimization plugin and illustrated the concept of topology optimization. However, a proper plugin for topology optimization may be required in the future, especially when adding more parameters.

The automated process of topology optimization constitutes the same challenges stated with component optimization. The main challenge is that input and output ports may differ in their type, and a proper ontology would be required to find generic transformation components or at least interchangeable components.

5.5 Conclusion

In this chapter, an exemplary implemented Genetic Algorithm based plugin for parameter optimization was integrated into the previously developed optimization framework. Developers may use this as an exemplary implementation for further plugins as it shows the usage of the proposed framework and API. The plugin did not have to deal with any Taverna internals that are required for example for the parallel execution or security mechanisms. Thus, the plugin integration was straightforward by implementing required methods for the API and focusing on the optimization algorithm and level in question.

The plugin makes use of a GA-library in order to implement the heuristic optimization process. The workflow parameters were encoded as genes and one parameter set as chromosomes, respectively. To take user knowledge into account, a panel was integrated into the optimization perspective in order to capture ranges for parameters and dependencies.

Four use cases taken from proteomics, ecological niche modeling, biomarker identification and structurally bioinformatics have demonstrated that the automated parameter optimization gives improved results compared to default values, manual search (trial and error) or parameter sweeps, while being more time and resource efficient. The proteomics use case leads to new findings that have been obtained after a second optimization process with higher parameter limits, which are not typically applied with the observed tools. These findings as well as the fact that three different aspects of this use case were optimized are a good way to inspire scientists to apply similar optimizations also to other proteomics use cases.

"The ecological niche modeling workflow was optimized as the default parameters provided non sufficiently accurate models. The improved SVM algorithm produced more valuable models that could then be used for further processing." [Obst2013]. The optimized biomarker identification workflow ranked all important genes higher than using the default parameters. Protein structure prediction optimization could gain a 10% better result with four times less executions compared to manually performed parameter sweeps.

The parameter optimization plugin was also used for component optimization and showed that component optimization is valuable.

The main contributions of this chapter can be summarized as follows:

- A parameter optimization plugin was exemplary implemented from the developer's perspective and plugged into the framework.

- Four different use cases were tested to evaluate the framework and plugin.

- Especially the Proteomics use case showed that parameter optimization is useful, as new insights of mass spectrometry were obtained.

- Suggestions and simulations were made for component and topology optimization use cases.

Chapter 6

Discussion: Scientific Workflow Optimization in e-Science

Within the presented thesis, the possibility to adapt optimization to scientific workflows was investigated. It was shown that there is a great potential to support scientists during the improvement of scientific experiments. The main problem is that there was no former solution available to automatically improve a scientific output of a workflow within a SWMS. Researchers could use the existing design frameworks but they often lack a graphical user interface and new models have to be implemented in a programming language, finally interaction with workflows is not possible at all. Commonly, researchers just adhere to the default or already proven good values. To improve a result, they used trial and error or parameter sweeps, both may be inefficient and time consuming. Within this thesis, a novel approach was developed and implemented in order to improve scientific workflows in an automated and generic manner.

This chapter will first discuss problems that can occur during the optimization process and presents approaches that can improve the overall optimization process in the future. Some of those approaches have serious requirements regarding a minimum of available executions or computing time. To overcome these requirements, the second part of this chapter will introduce a novel approach in order to integrate optimization meta-data into the provenance data of scientific workflows. This extended provenance data could be used to overcome many difficulties of workflow optimization in the future.

6.1 Examination of Scientific Workflow Optimization

Within this thesis, a subset of possibilities and improvements that can be achieved by using optimization instead of trial and error or parameter sweeps were discussed. This is due to the large variety of available optimization algorithms, various improvements of algorithms, different variations of algorithms, and so on. This section explores the field of optimization and investigates several approaches to improve the optimization of life science workflows. In the beginning, some general problems of optimization are embedded in the context of workflow optimization.

6.1.1 General Aspects of Optimization

General Optimization Issues of Heuristic Methods

Optimization in general and real-world problem optimization in particular can involve many risks of failures or problems. Weise et al. [Weise2009b; Weise2012] published a general overview of problems and pitfalls of optimization, especially considering evolutionary concepts. Some typical problems include among others the premature convergence, a non-effective landscape and multimodality, or epistasis.

Premature convergence can occur when applying small population sizes, especially in real-coded Genetic Algorithms (selected for the example plugin in this thesis) when selecting non-optimal natural operators (selection, crossover and mutation). The operators for Genetic Algorithms, such as One-Point crossover, are developed for binary-coded Genetic Algorithms and lead to an increased diversity when using them in real-coded Genetic Algorithms. Theoretically, each point in the search space must be reachable by only applying natural operators [Goldberg1990]. Therefore, adapted operators need to be used in real-coded Genetic Algorithms [Goldberg1990], which was done within this thesis implementation. These specific operators [Herrera1998; Herrera2003] can balance the diversity, exploration and exploitation like in binary-coded GAs. In general, this phenomenon is also known as genetic drift [Rogers1999] when the populations seek to a local optima due to minor diversity and missing selection pressure.

The *shape of the fitness landscape* is a severe problem. A fitness shape can have many properties such as i) multimodality, when many local optima occur, ii) ruggedness, when the fitness function is unsteady and produces a weak causality so that it is hard to select *'important'* regions, iii) deceptiveness and neutrality that occur with local minima or hyperplanes and means that local minima may be more attractive than the global optimum or hyperplanes do not provide any gradient information to reach the optimum. All these issues lead to the finding of local optima, no optima at all and slow or non-convergence. Certainly, the Genetic Algorithm may find a good solution some time, but this might take longer than expected, or at least a very large population is required. Within this thesis, some examples showed minor beginnings of those properties, like the Proteomics parameter optimization illustrated in Figure 5.7. This landscape has a comparatively large hyperplane and local optima.

Epistasis occurs if one gene influences the property of other genes. In real-coded Generic Algorithms this issue can especially occur if one gene activates or inhibits other genes. Like other issues, the impact can only be weakened by applying a larger population and diversity increasing operators.

All these issues can be addressed by using different natural operators (selection, crossover and mutation) and a relevant population size. The calculation of the '*correct*' population size is a very controversially discussed research topic [Michalewic1996, p. 72]. Whereas most researchers agree that a population size too *'small'* can guide the optimization to local optima and a population size too *'large'* can cause very slow convergence, there is no single solution available for this problem even if some claim there is [DiazGomez2007]. This fact, combined with the full parameterization of the optimization algorithms themselves, becomes a crucial issue for non-optimization experts. Currently the developed optimization parameter plugin includes adjustable parameters for crossover rate, mutation rate, number of generations and population size of the GA, which may still need to be adapted to the search space at hand. However, this is a general problem of heuristic optimization techniques and initializing these values could be supported by automated tools [Ochoa2012] or based on provenance in the future, as will be outlined below.

The natural operators might also be adapted during the optimization process. In doing so, the algorithm can tune the operators during the optimization process, in order to adapt the different significances of the operators to the respective stage of the process. For these self-adaptive Genetic Algorithms [Davis1989] several different approaches and many diverse algorithms exist [Palit2005, pp. 312 ff.]. Generally speaking, they aim towards the adaption of the genetic operators to a specific fitness landscape. This can be achieved for instance by adapting the mutation rate, crossover rate or selection strategy [Back2001]. Other approaches also target towards the adaption of the population size. Changing the crossover rate is the most common approach of an adaptive Genetic Algorithm. Some methods aim towards the calculation of weighted parameters based on statistics gathered from past generations. Others target towards the creation of a new offspring based on the calculated proportion of the difference between the two locations of the parents in the search space [Beyer2001]. However, other statistic based methods could be more successfully if a provenance-based system was available as will be described below. Statistics of other searches could then be integrated in the adaption process of operators. This method would be more successful since the landscape would be roughly known before the optimization process starts and operators could guide the optimization into more interesting regions.

Another approach is the usage of hybrid algorithms, as outlined in Chapter 2.2. Hybrid methods include two different search mechanisms during one search in order to utilize their different advantages. In principal, they are very successful during the exploration of global optima as they supplement a strategic area search around promising solutions after each generation. They are of advantage, if a convergence to high accuracy is required. In a later sub-section (Section 6.1.2), approaches for the handling and usage of hybrid methods for scientific workflow optimization are proposed.

Comparison to other Design Frameworks

Compared to other established design frameworks [Rao2009] (also refer to Chapter 2.2.3), which are used in the engineering domain, the developed framework within this thesis aims at providing life scientists with parameter and other optimization methods, who typically have experience neither in computer science nor in optimization. In conclusion and as published in [Holl2013f], the developed framework has various advantages over some engineering frameworks:

- The optimization problem can be defined by a workflow and not by an implemented script.
- Fitness functions can be formulated by any Taverna service (e.g. in an arbitrary programming language or Web service); for many common cases readily provided as Taverna components.
- Graphical user interface integration is provided within Taverna, a commonly used SWMS in the life sciences.
- Workflows require no changes in order to be optimized and therefore can be monitored, edited, and shared like any other Taverna workflow.

- The optimization process works in parallel and is therefore comparatively fast.

"The sum of these features should make scientific workflow optimization easy and fast enough to be adopted by life scientists.

At present, the workflow optimization framework cannot be compared to design frameworks in the engineering domain regarding the breadth in methods and more advanced features like multi-objective optimization [Deb2001] or hybrid optimization [Blum2011] (refer to Chapter 2.2 for more details). Many optimization methods are available in engineering design frameworks and could be ported into the developed workflow optimization framework. In order to allow a fast porting, an API that provides abstractions of all Taverna internals like security mechanisms, parallel execution and data handling on the Grid as well as GUI integration have been designed. This opens an easy way for developers to add further optimization plugins, which implement these more advanced workflow optimization techniques including multi-objective optimization. Due to the complex fitness landscapes of scientific workflows, other meta-heuristics such as Particle Swarm Optimization [Kennedy1995; Engelbrech2005] or Ant Colony Optimization [Dorigo1999] for continuous problems would be other useful additions supplementing the already presented parameter optimization plugin based on Genetic Algorithms." (Modified from [Holl2013f].)

Real-coded Heuristic Methods

The usage of a real-coded heuristic method is generally recommended when realizing parameter optimization [Wright1991] for real-world problems [Deb2002a] due to their closeness to the problem representation. Other preferences given in [Goldberg1990] are: i) comfort with one-gene-one-variable correspondence, ii) avoidance of Hamming cliffs and other artifacts of mutation operating on bit strings treated as unsigned binary integers, iii) fewer generations to population conformity. Especially the property of faster convergence, high accuracy, high performance and high precision [Michalewic1996, pp. 105,106] made this technique attractive for usage within this thesis as the optimization of workflows can and typically does require more execution time than common optimizations. Additionally, the runtime may be dependent on other factors such as server capacity or server robustness. Within the use cases experimentally tested in Chapter 5, a relatively small population size and fast terminating abort criteria was selected. Albeit the scientific output obtained within the four presented use cases have been significantly improved, a small population can cause the above described issues during the optimization process. The reason for choosing relatively small populations is mainly based on computationally reached limits. Nevertheless, these computing limitations are based on problems that occurred especially during the execution of several workflows in parallel. If the sub-workflow instances are executed in parallel on the client machine, the workload is large and often not feasible (for real-world applications). By moving the executions to distributed computing infrastructures, this problem was successfully solved. However, execution on the computing infrastructures does not involve unlimited computing budget or a suitable middleware that is able to manage the parallel execution of a large number of jobs. Especially the latter was identified as a challenge during this thesis project and a population size of 20 was determined as a reasonable trade-off between error rate during execution and already suitable population size. The population size was also partly influenced by the execution time of single jobs, one generation and the entire optimization process. Finally, the population size also depends on the problem at hand and on the search space available, as outlined previously. Another well-known problem that also occurred during the optimization of many individual workflows in Taverna is: '*java.lang.OutOfMemoryError: Java heap space*' [myGrid2011]. This exception is thrown if the required memory of the Java virtual machine exceeds the available memory and new objects cannot be allocated. Due to Taverna's system implementation this can happen from time to time, especially if a large number of Web service calls have been executed. Therefore, some effort should be put in the future into more robust software tools that can handle the complexity of workflow optimization.

Selection of an Objective Function

"An important issue in optimization is the selection of an appropriate optimization objective function. The most common optimization approaches in the life sciences use comparisons to a 'gold standard' and hence minimization of a single objective measure such as root-mean square deviation (RMSD) or a correlation measure. In many cases, finding a suitable fitness function may remain a challenge.

In contrast to the engineering domain, multi-objective optimization has not yet been widely used for parameter optimization in the context of life sciences [Sun2012], although it would be helpful to assess trade-offs, for example between sensitivity and specificity or between result quality and runtime. Additionally, the calculation of a Pareto optimal set would still be more cost and compute-intensive than the optimization of a single-objective problem [Deb2001, p. 46]. Many solutions exists, such as a weighted sum or ε-constraint method [Deb2001, pp. 50ff.], which allow combining multiple objective functions into one single objective. In some cases it may be sufficient to roughly approximate two objective functions by choosing a single objective measure with an explicit trade-off parameter such as F_β [Rijsbergen1979] and perform separate optimization for a few values of the trade-off parameter (β) such as runtime and original fitness measure. For example the squared correlation coefficient in use case 5.3.4 could be substituted by an F_β component." (Modified from [Holl2013f].) This extended fitness function with two objectives can be integrated in the proposed workflow optimization framework. As the fitness function is described by a component in the workflow, changes and adaption such as the addition of F_β can be conducted by the user. Another approach would be to use a weighted objective function for instance in the extended use case for Proteomics in Section 5.3.1. The runtime of the workflow was six-times longer adding a certain parameter and selecting a certain parameter value. However, the parameter adding resulted only in a marginally better solution and was therefore not recommended. Adding the runtime as another fitness value would had lowered the fitness value for the specific parameter value that causes the longer runtime.

As outlined in Chapter 2.2, multi-objective optimization methods aim at the calculation of a so-called Pareto front. The Pareto front represents all possible solutions that satisfy possible trade-offs between the objective functions. Multi-objective optimization methods, like the NSGA-II [Deb2002b] can be integrated into the proposed workflow optimization framework by extending the proposed API. Due to complexity issues, multi-objective optimization was not taken into account during this thesis. Nevertheless, in the future it would be required to solve problems that are even more complex. Thus, the next section discusses some approaches for approximation and the feasibility in workflow optimization. Most of these approaches can also be applied to single objective optimization [Jin2005] as well as hybrid methods and could thus generally be attractive for workflow optimization.

6.1.2 Addressing Optimization Complexity

Due to the concept of multi-objective optimization, a much larger search space has to be searched in order to find the Pareto front. For computationally expensive or compute-intensive fitness evaluations, this large number of required evaluations represents a real problem [SantanaQui2010]. One approach are mechanisms that evaluate the fitness functions in parallel. However, the highly required inter-connection for sharing the *non-dominated* solutions can be challenging by means of a framework [Alba2013]. The framework presented within this thesis is already able to compute the fitness evaluation in parallel by shifting the workflow execution to distributed computing infrastructures and executing single processing steps on high performance computing systems. In the same manner multi-objective optimization fitness evaluations could be executed, as the framework manages this parallel execution, and plugins can leverage the API to reuse the available parallel submission mechanism. However, parallel execution cannot solve the complexity itself. Therefore various approximation methods have been approached within the last years [Neri2012; Tenne2010; Deb2001]. *Approximations* occur in many fashions and all of these methods can be used for single-objective methods similarly as for hybrid methods or multi-objective algorithms [Jin2005] in principal.

Some approximation strategies are: the reduction of the design space to explore only *interesting* regions; screening (sensitivity analysis) of significant variables; mapping or visualizing the design space, so that the user can play a significant role for further exploration strategies [Shan2010]. Additionally, many efforts have been and are recently being put into research of approximation and active learning in order to predict design models [Jin2005; Shan2010; Tenne2010; Di Nuovo2012; Zuluaga2013].

Parameter Sensitivity analysis

The screening of significant variables [Shan2010] can determine and reject unimportant input variables and keep only the important ones for optimization in order to reduce the design space. In doing so, dependencies and interactions among different variables and the problem have to be analyzed. This is often implemented by sampling a certain amount of test points. The analysis of one fitness function can then be performed for example by using sensitivity analysis or analysis of variances (ANOVA) [Shan2010]. The sensitivity analysis identifies important variables regarding to their changes and following variability of the output at a specific point. This variability can be measured for single variables or for all variables at the same time. ANOVA identifies important variables by analyzing individual combinations of variables. In doing so, variances of the individual dependencies and the overall dependencies can be measured. Both and also other methods for parameter significance analysis are relevant for workflow optimization, as users may not be aware of all important parameters or include unimportant parameters into the optimization process. The developed workflow optimization framework could perform the analysis during the optimization process and eliminate identified unimportant parameter from the search space automatically. Another more advanced and significant approach would be to store analysis information within optimization provenance. Optimization provenance will be outlined later in this chapter, and at the moment it is sufficient to know that there could be optimization provenance storing on the one hand results of prior optimization runs or on the other hand already performed sensitivity analysis meta-data. The framework could take these prior executions into account to perform the significance analysis and present the important and unimportant parameters to the user *before* starting an optimization run. Accordingly, the user can redefine and reduce the search space manually. At the same level, this meta-data can be analyzed for higher level optimization (component or topology). In case of topology optimization, a certain switching of components may not have any impact on the final result and can thus be omitted. Finally, one should note that the removing of variables reduces the search space but might also cause removed accuracy.

Space reduction

Another approach is space reduction by omitting non-relevant regions in the design space. This is achieved by different methods that predict good regions in the design space, eliminate individuals that would potentially lie in worse regions and thus guide the iterative sampling process. One new approach is the usage of so-called probabilistic model-based Evolutionary Algorithms [Lozano2009] (EAPMs). These algorithms do not use any crossover or mutation to generate new populations, instead they use a probability distribution model consisting of global information statistics gained from previous searches. Based on these models, only individuals in promising regions are produced and evaluated. Another approach is based on the measurement of neighboring and parent fitness values [Triguero2012]. This method selects new individuals depending on the fitness value of their parents or neighbor individuals. Furthermore, another method suggests to pick the next sample based on the predictive uncertainty of Bayesian models or Gaussian processes [Zuluaga2013]. In a similar manner, SVMs could be used to classify parameters based on their fitness values and thus eliminate non-promising individuals. All these methods would nicely fit into workflow optimization due to the reduction of required fitness evaluations. Unfortunately, such predictions can only be made if a certain number of evaluable points are available. A serious prediction could be produced using prior workflow optimization runs, as roughly explained within the last topic and outlined in the next section. Based on already performed optimization runs, these learning approaches could predict valuable points right from the beginning. These methods could also be used to predict an initial population based on prior runs, which would mainly reduce the problem of a relatively small population size [Reeves1993], which was mentioned as a problem above.

Visualization of the design space

One of the most straightforward methods is the visualization of the design space or the obtained results [Takagi2001]. Then the user would be in control of the evaluation of good and bad regions in the designer space. Especially within multi-objective optimization, this technique might be very helpful for reducing the search space. As the framework keeps track of statistics already during the optimization process, these could be visualized during the optimization process to receive user feedback on more interesting or uninteresting search space regions.

Surrogate Models

Different approaches have been developed in order to replace the compute-intensive function with a surrogate model. Surrogate models can approximate the behavior of i) the global problem, or ii) several models that approximate local –area specific– problems [SantanaQui2010], to solve the fitness function at low cost. During an optimization process, several fitness values can then be approximated and only a few remaining fitness values are calculated. Commonly used methods to design surrogate models are the Response Surface Methodology, Radial Basis Functions, Artificial Neural Networks, Kriging, Gaussian Processes or Support Vector Machines, which are summarized and described for instance by Santana-Quintero et al. [SantanaQui2010]. These approaches themselves are targeted at many research activities as they demand further optimizations. Additionally, they require a certain amount of reasonable available test points, which can become a crucial factor as described earlier.

Further approaches aim at knowledge incorporation and improve the surrogate model(s) during the optimization processes by learning from previous fitness evaluations [Di Nuovo2012]. However, generating approximation models is neither straightforward nor is the approximated function globally correct due to high dimensionality.

For hybrid methods, approaches exist that suggest the usage of surrogate models for the local or global search. Both approaches present a convenient trade-off between calculated and approximated fitness values. In principal, the described approximation approaches could be adapted to scientific workflows and integrated in the optimization framework. This integration could follow the described implementation of the API methods (cf. Chapter 5.2). To execute the surrogate models, either a specific workflow could be created that is then carried out by the framework, or the plugin runs the small piece of code on its own. However, from the developer's point of view, the first approach should be favored. Certainly, these approaches follow the other proposed techniques and require a certain amount of test points. Likewise, they could give their best expression with the integration of provenance meta-data, which will be discussed in detail within the next section: learn from prior workflow optimization runs.

6.2 Provenance-based Optimization

As stated in the last section, the usage of further workflow optimization methods requires more sophisticated techniques to ensure sufficient sampling of the search space in a reasonable amount of compute time. Apart from user constraints and approximation methods, any information obtained from prior optimizations of similar workflows might be useful to guide and improve the optimization process. At present, the optimization process for each workflow needs to be performed from scratch and is neither captured nor shared among scientists. In the following section a novel approach to capture workflow optimization meta-data is proposed and different scenarios are outlined.

Workflow definitions and execution traces can already be captured locally by workflow management systems and it would generally be possible to store additional information such as optimization meta-data. While currently, community repositories only store workflow definitions, performance measures and other meta-data of optimization runs etc. could be stored likewise. If such an e-Science infrastructure for information-enriched workflows would be available, other techniques could apply this information for optimization.

As an example, already optimized workflows could guide a new optimization process: During parameter optimization, an optimization method could initialize a part of the first population with parameter values which have already been identified as optimal/best. In doing so, the optimization process could directly start to sample the more promising search space regions. Another example is to store surrogate models in optimization provenance in order to reuse and refine them with each optimization run. By doing so, the approximation of a specific scientific experiment can be improved step by step by the community.

Workflow optimization provenance data could also be used to search and especially rank similar workflows for a specific scientific problem. By now, search results can be sorted by manually added ranking scores or by the *'Most viewed'* or *'Most downloaded'* rate within myExperiment [UoM2007]. With the available optimization provenance data, further ratings focusing on the real scientific output could be addressed. A ranking system could analyze already performed optimization runs of a specific problem at hand and compare for example i) number of available optimization runs, ii) obtained fitness values, or iii) similarity of input data. A top ranked workflow from case i) would reveal that it is a robust optimized workflow with high performance inference. A further optimization run may therefore be significantly reduced in search space dimension or just omitted. With a top ranked workflow from case ii), the user can decide whether the fitness is sufficient and make a test run with his own data set or rerun an optimization with the best fitness values as a starting population. A top ranked workflow from case iii) would reveal that the data input was most akin to the data set at hand and thus might be already optimized for this data set. A user could then just reuse the workflow and optimized parameter set. Certainly, these three cases would and should be able to be combined in the execution. For example, the highest ranked workflow has been optimized several times, has the best fitness value, and the used data is the most akin to the data set at hand. This workflow would reflect the *best practice* and the user might not pursue any further optimization steps.

Although such a decision support system and provenance-based optimization in general could be established in larger local environments, the full potential would be utilized by a community wide adoption. For a collaborative usage, optimization provenance must not only be sharable and reusable, but also reproducible. To implement these functions, SWMSs or portal solutions need to capture optimization runs and interpret optimization provenance data. Thus, a novel approach is required that provides a model to store, share and reuse relevant optimization entities. In order to adopt optimization in e-Science infrastructures quickly, this model needs to be based on an already existing solution instead of developing a method from scratch. A possible approach could be based on the model of so-called *Research Objects (ROs)*, which are described in the following.

Research Objects

While the workflow repository myExperiment [UoM2007] was initially aimed towards workflows, the novel development version of myExperiment[UoM2012] aims at complex forms of sharing, reuse and preservation and has been extended to incorporate the notion of so-called workflow-centric Research Objects [De Roure2009a; Belhajjame2012a]. Workflow-centric Research Objects, i.e., packs in myExperiment, have recently been developed as one aim of the wf4ever Project [UoM2011]. They aggregate different kinds of resources including the workflow specification, annotations that describe the workflow, provenance traces, or others and have been described in [Holl2013c] as the following:

"Research Objects aim at providing support for the description of scientific investigations in a machine readable format. In addition to the scholarly article that reports on the results of the research investigation, a Research Object encapsulates other resources that enable and promote the reuse, interpretation and reproducibility of such investigation results. In particular, a Research Object comprises the datasets used and generated during the research investigation, the workflow encoding the experiment carried out, the provenance traces captured by running the experiments and the various annotations that describes resources and their relationships. Workflow-Centric Research Objects, i.e., Research Objects aggregate a number of resources, namely:

- A *Workflow*, which defines a template with the interconnected series of steps necessary to specify a given experiment.
- The *WorkflowRuns* obtained by executing a Workflow in a Workflow management system.
- The *Annotations* used to describe the current Research Object, its resources and their relationships.
- Other resources used to help providing context to the research investigation, for instance a paper describing the research, the hypothesis of the experiment, bibliography related to the experiment, conclusions and interpretations of the results, configuration files, etc.

The Research Object model is represented by a family of ontologies [Belhajjame2012b], which is divided into a Research Object Core Ontology [Wf4Ever Pr2012] and extension modules that cater for different domain specific requirements (Research Object evolution, scientific workflows, etc.). The Research Object Core Ontology and the ontologies used to specify Workflow-Centric Research Objects: the *wfdesc* ontology [Wfdesc2012] (used to specify workflow templates) and the *wfprov* ontology [Wfprov2012] (used to capture the provenance traces of the workflow executions)." [Holl2013c]. As the current Research Object model is not applicable to store optimization provenance, the model was extended to capture optimization runs and is described in the following.

Research Object Optimization Ontology

Many relevant objects are created during an optimization run and need to be stored in a structured way. For reuse, not only optimization meta-data are required, such as the used algorithm, the termination criterion or the defined search space, but also the obtained fitness values. Therefore, a novel Research Object optimization ontology *(RO-Opt)* was designed to capture all relevant entities, produced during an workflow optimization run. This ontology was published in [Holl2013c] and is presented in the following.

The ontology concept is shown in Appendix A.14 and maps the optimization entities to Optimization Research Objects. "An 'Optimization Research Object' [Holl2013b] (opt:OptimizationResearchObject) is an aggregation of all the resources required for performing the workflow optimization process presented in Chapter 5. Since 'Research

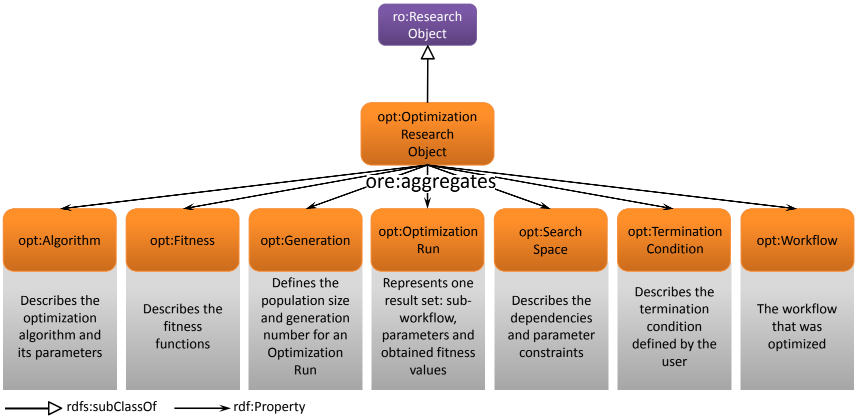

ro:Research Object
opt:Optimization Research Object
ore:aggregates
opt:Algorithm
Describes the optimization algorithm and its parameters
opt:Fitness
Describes the fitness functions
opt:Generation
Defines the population size and generation number for an Optimization Run
opt:Optimization Run
Represents one result set: sub-workflow, parameters and obtained fitness values
opt:Search Space
Describes the dependencies and parameter constraints
opt:Termination Condition
Describes the termination condition defined by the user
opt:Workflow
The workflow that was optimized
rdfs:subClassOf
rdf:Property

Figure 6.1: The Research Object Ontology developed to store optimization provenance.(Source: [Holl2013c])

Objects' (ro:ResearchObject) are aggregations of the resources used or referenced in an investigation, an Optimization Research Object is a specific type of Research Object. Figure 6.1 shows an overview of the main resources aggregated as part of an Optimization Research Object: The 'algorithm' (opt:Algorithm) used to generate the optimization parameters, the 'fitness function' (opt:Fitness) used, the population of individuals (solutions) that have been produced ('optimization run' (opt:OptimizationRun)) at a given iteration of the optimization algorithm ('generation' (opt:Generation)), the search space (opt:SearchSpace) where the algorithm has searched for optimum values, the 'termination condition' (opt:TerminationCondition) for the optimization algorithm, the 'workflow' (wfdesc:Workflow) being optimized and the 'link to the best results' (opt:hasBestResult) found. In the following subsections each of these aggregated resources is described.

Algorithm

As the fitness landscape of scientific workflows can be rugged and contain no gradient information, meta-heuristic search methods are convenient search algorithms (opt:Algorithm) to deal with the optimization of scientific workflows. Appendix A.10 shows the type of search algorithms, e.g Genetic Algorithms (opt:GeneticAlgorithm), Particle Swarm Optimization (opt:ParticleSwarmOptimizationAlgorithm) or other methods that are conceivable and can be implemented as a plugin to the optimization framework.

Fitness

In order to automate the optimization process, workflow results are evaluated by the optimization algorithm. The workflow result is based on an output parameter (opt:FunctionOutputParameter), which represents a specific fitness measure (opt:Fitness). Appendix A.11 depicts an overview of the fitness section of the ontology. The fitness measure may consist of one or several output parameters. Several parameters and weights are used, if the user wants to perform a multi-objective optimization (opt:MultiObjectiveFitness) instead of a single-objective optimization (opt:SingleObjectiveFitness). The fitness function (opt:FitnessFunction) associated with the fitness measure can not only store output parameters and weights, but also a body (opt:hasBody) which may contain a piece of code representing a unique measure description.

Generation

During the optimization process, each workflow instance from a population of size y is executed. Each of these instances represents a unique parameter and component combination. After the execution and evaluation of the workflow instances, a new generation of unique workflow set-ups is executed.

To monitor the population evolution, each individual workflow run has a corresponding generation number (opt:hasGenerationNumber). Additionally, the population size of each generation is stored (opt:hasPopulationSize), as this number can vary for optimization runs in general and each generation in particular.

Optimization Run

Results obtained during the optimization process (opt:OptimizationRun) may be of interest and it is necessary to learn from them in later optimization runs. Thus the executed workflow instances and their results should be captured. It is recommended against storing the entire provenance trace of a workflow run as typically the original trace has already been saved. This is due to the fact that many relevant data objects (e.g. intermediate results) are stored with the provenance trace to achieve reproducibility. It can be assumed that these data objects are negligible when achieving optimization reproducibility and especially when learning from optimization runs. The crucial values to capture are fitness values (opt:hasFitnessValue) and a flag (opt:Flag), which depicts the origin of the fitness value. As shown in Appendix A.12, the fitness value may be calculated (opt:Original), approximated (opt:Approx) or just taken from a prior similar or identical optimization run (opt:LinkToOriginal).

Search Space

Each workflow adaptation represents one specific set-up and can be sampled in a specific search space (opt:SearchSpace). The search space is spanned by the selected workflow parameters and/or structural changes. As not all parameter values or combinations of components are valid, the number of tested workflow set-ups can be reduced by intelligently sampling the search space by an optimization algorithm. The dependencies of the search space should be stored in order to allow other optimization processes to reuse the search space later. As shown in Appendix A.13, the search space comprises component and parameter dependencies (opt:ProcessorDependecy and opt:ParameterDependency) respectively as well as parameter constraints (linked with the data property opt:hasParameterConstraint). Dependencies can be asserted with the opt:onParameter and opt:hasTargetProcessor properties.

Termination Condition

The optimization process will stop at a certain time – preferably when the algorithm has converged. However, as this may be a heavy time consuming task, and thus the user often wants to add additional termination criteria. The termination condition (opt:TerminationCondition) should be stored due to reasons of the plausibility of the optimization run. It precisely stores under which condition the optimization is to be terminated. This can be the maximum number of performed workflow executions (opt:hadMaxNumberOfExecutions), the maximum number of evolutionary steps (opt:hadNumberOfSteps), the time required for the optimization process (opt:hadMaxTime), a reached fitness value (opt:hadFitnessReached) or a number of generations that did not improve the fitness measure (opt:noChange). Depending on the workflow, the search space settings and how strict the termination condition has been set, the influence on the value of the recorded best result can be ascertained.

Workflow

The original workflow (wfdesc:Workflow) should be stored in the Optimization Research Object to increase the optimization process discoverability (with the opt:hasWorkflow property). The workflow may be stored in an abstract or concrete fashion to capture the scientific experiment. Together with this resource, one representative workflow run (wfdesc:WorkflowRun) should be stored to capture all used input data.

Link to the best results

To ensure a fast and easy analysis, link(s) to the best result(s) (opt:hasBestResult) should be stored. For single-objective optimization one link is stored, for multi-objective-optimization several results representing the Pareto Front are stored." (Modified from [Holl2013c].)

In order to demonstrate the creation and storage of optimization provenance data, a part of the optimization run of *Prorocentrum minimum* via the SVM as demonstrated in Chapter 5.3.2 was exemplary captured with the RO-Opt ontology and published in [Holl2013c]. "In doing so, the described ontology was applied manually to model the performed optimization process as an Optimization Research Object and thus to create optimization provenance. This model stores the three parameters modified during the optimization run and their constraints. The workflow instances as well as the best results have been recorded. Appendix A.15 shows a fraction of the RDF encoding the example. In particular it captures how the optimization run is modeled and shows the record for the best fitness obtained during the optimization run. The Optimization Research Object is available online [Holl2013a].

As optimization meta-data can now be stored in a structured format, it can be read and interpreted by a program and by a scientist alike. Even a single researcher can already benefit from this meta-data provenance, as the optimization algorithm could include the results during a second, third, and other optimization runs of the same workflow optimization. The algorithm may thereby identify gradient information and converge to a (local) optimum more quickly. In addition, meta-data of the configured optimization algorithm or adaptive genetic operators could learn not only from the current optimization but also from prior optimization runs.

Workflow optimization and the presented ontology would benefit from a community wide adoption to utilize their full potential. Researchers may want to share their Optimization Research Objects, analyze results from colleagues and reuse their search space constraints, fitness definitions, and so on. The wf4ever project goal is to equip the myExperiment infrastructure with Research Object support and Research Object related services [Page2012]. In almost the same manner, the developed optimization Research Object based concept may be used in the future. This would enable users to access (and possibly query) the repository to find out, amongst other things:

- Which parameters have ever been used as part of the optimization?
- Which parameter combination performed best for a given workflow?
- Which parameter ranges have been used during prior workflow optimization runs?
- Which was the best result ever obtained for a specific workflow?

- Which parameters seem to have larger influence on the final result?

These are only some questions that could be answered by such a repository." (Modified from [Holl2013c].) Further questions and optimizations could also target workflow components, optimization algorithm parameters or surrogate models as outlined in Chapter 6.1.

If SWMSs would be able store optimization Research Objects in the future, they could automatically transfer annotated Research Objects to the myExperiment platform. The proposed optimization framework could be extended, in order to download related research objects from myExperiment and forward the extracted values to the individual plugins. The plugins may then be aware of how to reuse the additional information.

This approach aims at the capturing and reuse of optimization provenance data in SWMSs or portal solutions, according to the proposed ontology. Consequently, a SWMS would have a predefined list of entities to store and entities that can be reused. However, other optimization provenance concepts may be developed and it may be required to define a minimal semantic annotation model for optimization in the future. Similar to the available minimum information models for output data, such as the Minimum Information About a Microarray Experiment (MIAME) [EBI2010; Brazma2001] or the Just Enough Results Model (JERM) [Bechhofer2013; BBSRC2013], a minimum information checklist may be required for workflow optimization. Especially when including regular workflow runs with workflow optimization it will be necessary to define, which meta-information is available from a corresponding Research Object. For example, a regular optimization run can neither provide information about a search space nor about a used optimization algorithm. However, these runs may be useful to perform parameter analysis. One possible option would be to utilize the Minimum Information Model (MIM) vocabulary and framework for scientific linked data [Gamble2012]. This approach aims at providing a generic mechanism to describe the minimum information required. The description can be defined in RDF format and is thus machine readable and interpretable. Other optimization provenance concepts could then produce minimum information conform optimization provenance Research Objects.

The approach of provenance-based workflow optimization offers many different possible scenarios to improve the process of workflow optimization. Provenance-based optimization can ease and improve the general optimization process of scientific workflows in the future.

Chapter 7

Conclusion

7.1 Summary of the Work

The reuse of workflows still represents a challenge for researchers, as they have to pass through several design and parameterization steps. Due to the rich variety of options, a scientist cannot have the required expert knowledge for each application for selecting and parameterizing them properly. Thus, the commonly used method is applied to incorporate default parameters or to stick to the parameters that were once evaluated as adequate. Further improvements of the scientific output are achieved by trial and error or execution of more sophisticated parameter sweep jobs. These four techniques may typically not provide an optimal scientific output and also become time consuming or compute-intensive respectively.

Moreover, utilizing and reusing scientific workflows effectively, scientists lack an efficient solution to improve scientific outputs in an automated and simplified manner. Such a method could not only provide the optimization of different workflow optimization levels by using optimization algorithms, but instead also offer services for learning from provenance of workflow optimization. These challenges were addressed in this thesis by study, design, development, examination and further extensions of a method for scientific workflow optimization and it was shown that a general approach for scientific workflow optimization is feasible.

The main contributions of this work are summarized and visualized in Figure 7.1, which shows the originated design concept that was formulated and implemented within this thesis to assess a novel approach that provides scientists with an automated method to improve the result of an arbitrary scientific workflow. More precisely, a novel scientific

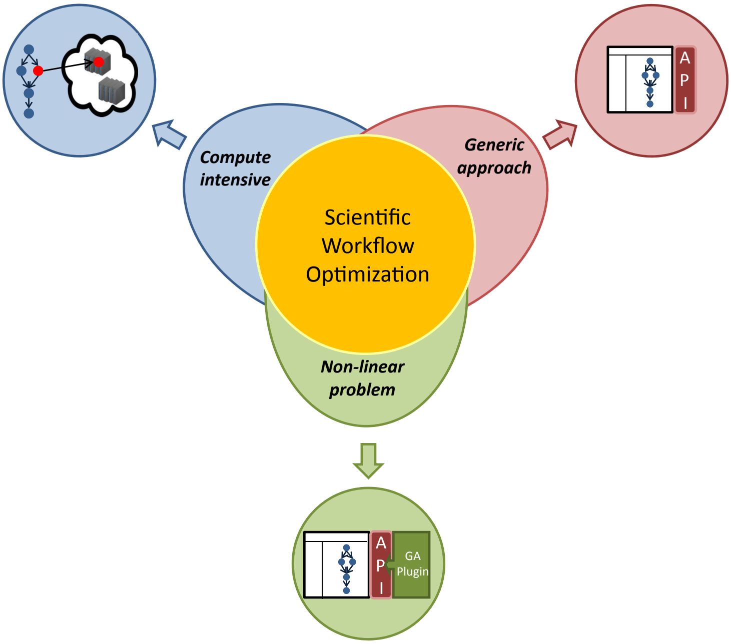


Figure 7.1: The three tier concept for workflow optimization including all implementations.

workflow life cycle phase was presented to optimize scientific workflows. The proposed phase concatenates with the common scientific workflow life cycle and augments the entire development process of a scientific experiment. By inserting this phase before the entire workflow is being executed, the optimization phase can serve as a substitute for several rounds of the complete scientific workflow life cycle by only employing a sub-workflow. Within this optimization phase, different levels of the workflow description can be target of the optimization process. This thesis has described parameter optimization, component optimization and topology optimization as possible optimization targets.

The Taverna workflow management system was chosen as the appropriate system for the life science domain (cf. Chapter 3). To implement a fast and easy adaption mechanism of workflow optimization by researchers, Taverna was used as a reference implementation within this thesis. In order to achieve acceleration and scaling of workflow execution, the usage of distributed computing resources accessible via e-Science infrastructures was implemented as a plugin in Taverna. This plugin enables the submission of workflow tasks or the entire workflow to a distributed computing infrastructure via the middleware

UNICORE. The performance of this plugin was demonstrated by a use case taken from the proteomics domain.

A novel open source optimization framework was developed to extend Taverna. The framework provides a new window in the Graphical User Interface and Taverna specific methods for security, parallel execution and data handling. The framework offers a generic application interface, which allows for easy integration of arbitrary optimization methods by developers. Accordingly, novel optimization techniques can be immediately provided for usage. The framework was extended further with a parallel workflow execution mechanism to lower the workflow load on the client during the execution of several workflows in parallel. This extension was experimentally tested by comparing the original computational approach with the developed parallel execution model. For proving the concept, the framework API was leveraged and a workflow parameter optimization plugin, implementing optimization via a Genetic Algorithm, was elaborated.

In addition, this developed workflow optimization tool was experimentally tested in four real-world life science use cases. All provided improved scientific results after the optimization process. The optimization of one of the use cases taken from the proteomics area provided insights into data and algorithms that had not been reported in such a manner until now. Additionally, a biomarker identification workflow was successfully optimized, which will be part of the planned IMI-project AETIONOMY [Fraunhofer2013].

Finally, the potential of future community repositories and ways of harnessing previous optimization runs that guide advanced optimization strategies have been outlined. The approach is based on Research Objects (ROs) that can store various provenance information of scientific workflows and the thesis proposed a possible extension to store workflow optimization meta-data. By storing these optimization Research Objects, further and more advanced optimization methods could learn from prior runs by reusing optimization provenance data intelligently to set up an optimization process or apply approximation methods.

The technical accomplishments of this thesis can be outlined as follows:

- A design concept was developed to investigate scientific workflow optimization.
- The concept was implemented by a framework for arbitrary workflow optimization using the developed parallel execution mechanism.
- The framework is generic as well as extensible and an example plugin was implemented and experimentally tested.

- Tested use cases produced improved scientific results after optimization and lead to new insights.
- A novel approach for the storage and intelligent use of optimization provenance meta-data was presented.

7.2 Future Work

This thesis has investigated the general approach of automated optimization methods of scientific workflows in e-Science infrastructures. The developed approaches may be extended in different ways for example by other optimization algorithms, higher level optimization techniques or advanced provenance-based optimization.

As this thesis is based on four diverse use cases, many other use cases are certainly conceivable. As briefly outlined in Chapter 5.1, particle swarm optimization, ant colony optimization or other optimization algorithms could be used to optimize scientific workflows. Additional optimization plugins, similar to the developed parameter Genetic Algorithm based plugin, implementing any optimization algorithm may be of benefit, to adapt the algorithm to the optimization problem at hand.

Component optimization as experimentally tested in Chapter 5.4 could be a further approach. For component optimization, a comprehensive ontology would be required to describe which algorithms or sub-workflows perform equivalent functions. Some domain specific ontologies have already been developed. Further studies could assess which ontologies are useful in combination with component optimization and whether these ontologies could be used in a collaborative manner.

The Research Object model for optimization is a current research topic and may be extended and improved in the future. A next step would be to integrate provenance data obtained from optimization runs in Taverna into a provenance repository, as it was recently initiated for simple workflow runs [Belhajjame2013]. With the initiation of such a repository, several questions from the user could be answered, such as "What was the maximum fitness ever reached?" or "Which parameter ranges have been successful for this workflow/application?".

Appendices

Appendix A

Additional Figures and Descriptions

A.1 Additional Figures of the Proteomics Use Case

A.1.1 Hybrid *E. coli* sequence logo

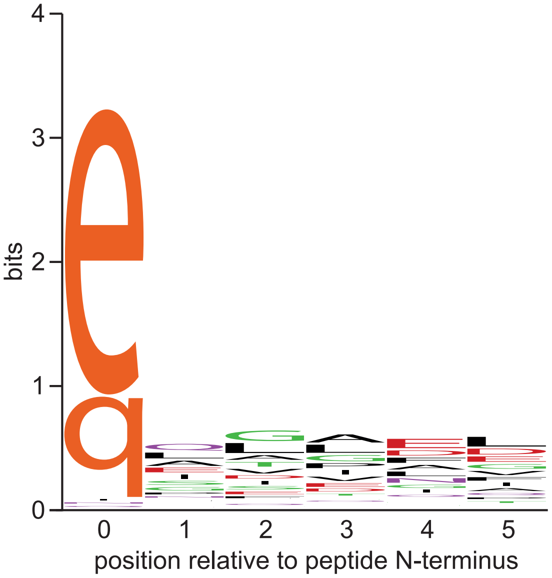


Figure A.1: Logo of the protein sequences between 16 Da and 21 Da from the hybrid *E. coli* data set. It shows the pyro-glutamate from glutamate and glutamic acid.

A.1.2 Peptide Identifications

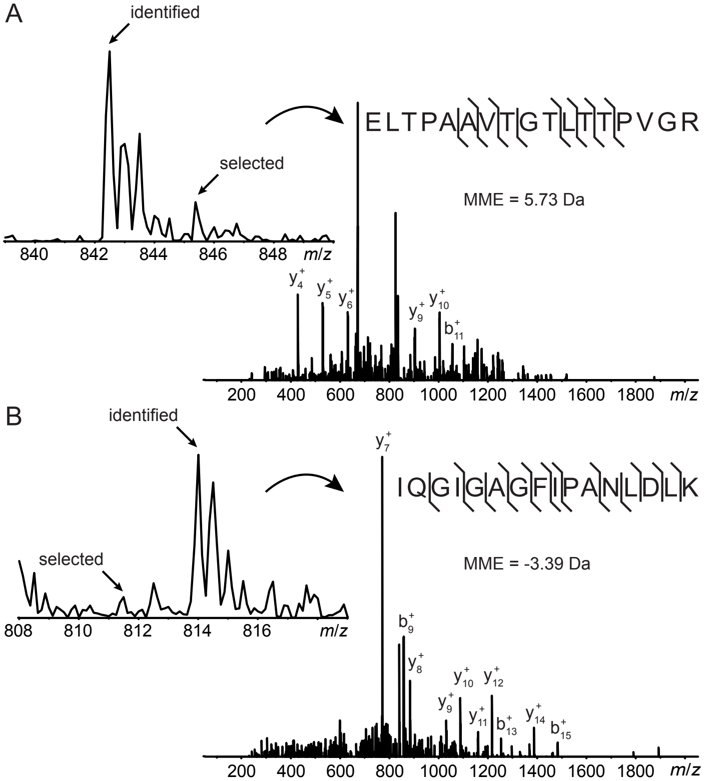


Figure A.2: "Peptide identifications from non-selected precursors. With larger MME tolerances, here ± 5 Da plus isotope error, X!Tandem identified co-eluting peptides with lower (A) or higher (B) m/z than the selected precursor but within the precursor selection. In A, a peptide with monoisotopic m/z 842.5 was identified (with PeptideProphet probability p = 0.989) instead of a selected peptide (or signal) at m/z 845.3. In B, a peptide with m/z 814.0 is identified (with p = 0.992) instead of a peptide at m/z 811.5. Both selected precursors were the ninth to be acquired out of ten sorted by intensity for their corresponding MS scans, more than 1.5 seconds after the MS scans themselves, and both precursors disappear into the background in the subsequent MS scans. These are two examples of almost 100 such peptide spectrum matches in the *E. coli* ion trap dataset." (Modified from [Holl2013e].)

A.1.3 *E. coli* sequence logo

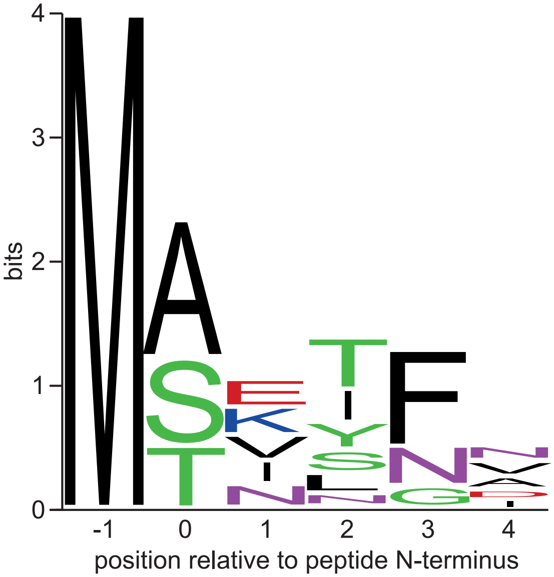


Figure A.3: Logo of the additional identified peptides. It shows the known pattern for cleavages of the N-terminal methionine.

A.2 Additional Figures of the Ecology Use Case

A.2.1 Description of the test species

"***Harmful algae blooms: Prorocentrum minimum (Dinoflagellata)***:
The increase in frequency and intensity of Harmful Algae Bloom (HAB)s has led to increased incidence of shellfish poisoning, large fish kills, and deaths of livestock and wildlife, as well as illness and death in humans. The economic repercussions of algae contamination can be very serious. Not only is fish production affected, through stock destruction and consumer mistrust, but there are also ramifications for the tourism sector. Although toxic algal blooms represent a serious public health and economic problem, no comprehensive forecasting systems for HABs is in place for research and management. Here we are testing the parameter optimization in the ENM workflow for a data set on the toxic invasive algae species *Prorocentrum minimum*. The data set contains 173 occurrence records of the species in the North East Atlantic, a geographic mask for the same region, and a set of environmental layers that drive the distribution of the species (Mean sea surface temperature, Mean salinity, Mean Photosynthetically Available Radiation).

Invasive species: The Pacific Oyster:
The Pacific oyster *Crassostrea gigas* (Bivalvia, Mollusca) originates from the Pacific Ocean, but has been the subject of widespread introductions either to replace stocks of indigenous oysters severely depleted by overexploitation or disease, or to create an industry where none existed before. The most significant introductions in Europe have been to France in the mid 60's. In addition to active and deliberate transfer of large number of individuals, Pacific oysters have also passively spread with long-distance vessels. Through natural spat-fall the species has dispersed and established feral populations on almost all continents with sometimes-large ecological and socioeconomic consequences. During the last decade, massive invasion has occurred in Scandinavia. Its establishment and dispersal may conflict with commercial, recreational, and conversational interests. At the same time, the species may contribute to important ecosystem services such as relieve algal blooms through their highly efficient water filtering ability, or serve as an important resource for commercial and recreational fishing. Here we are testing the parameter optimization in the ENM workflow for a data set on the Pacific oyster containing 121 occurrence records in coastal European environments, a geographic mask for the same region, and a set of environmental layers that drive the distribution of the species (Mean annual distance to

land, Maximum depth, Mean annual surface salinity, Mean annual surface temperature, Mean annual sea ice concentration)." [Obst2013]

A.2.2 ENM workflow with SVM

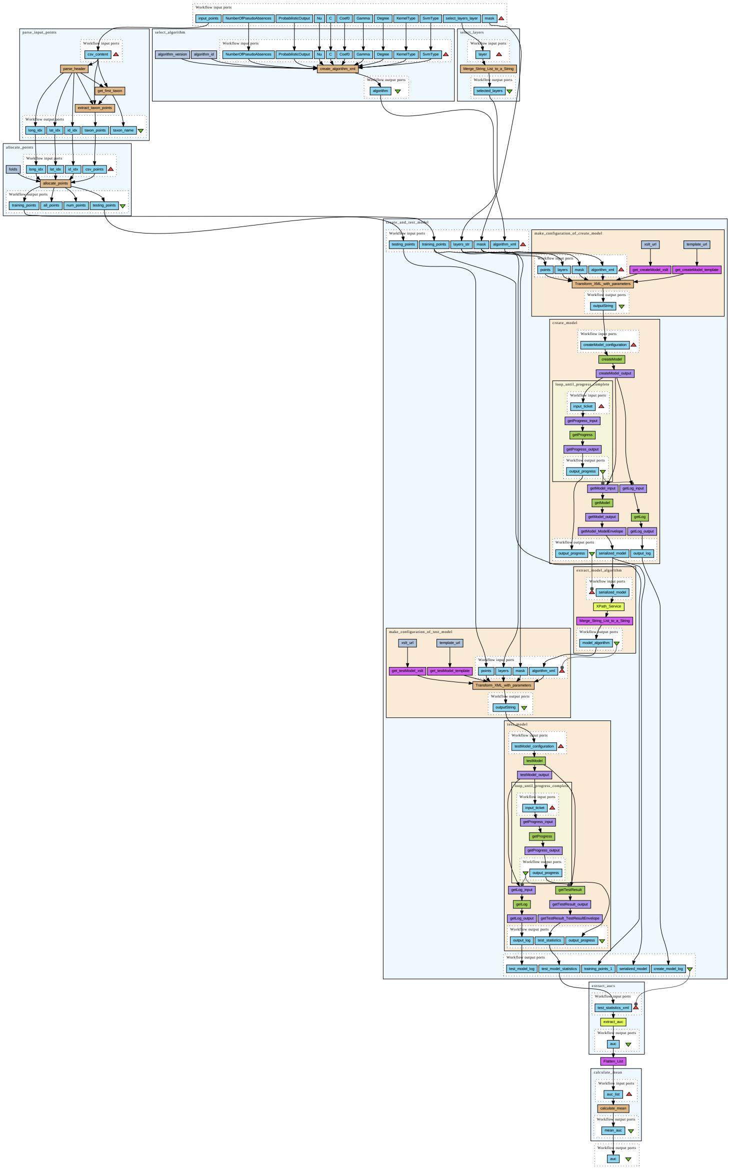


Figure A.4: ENM workflow for optimizing a support vector machnine (SVM). The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3680.html.

A.2.3 ENM workflow with Maxent

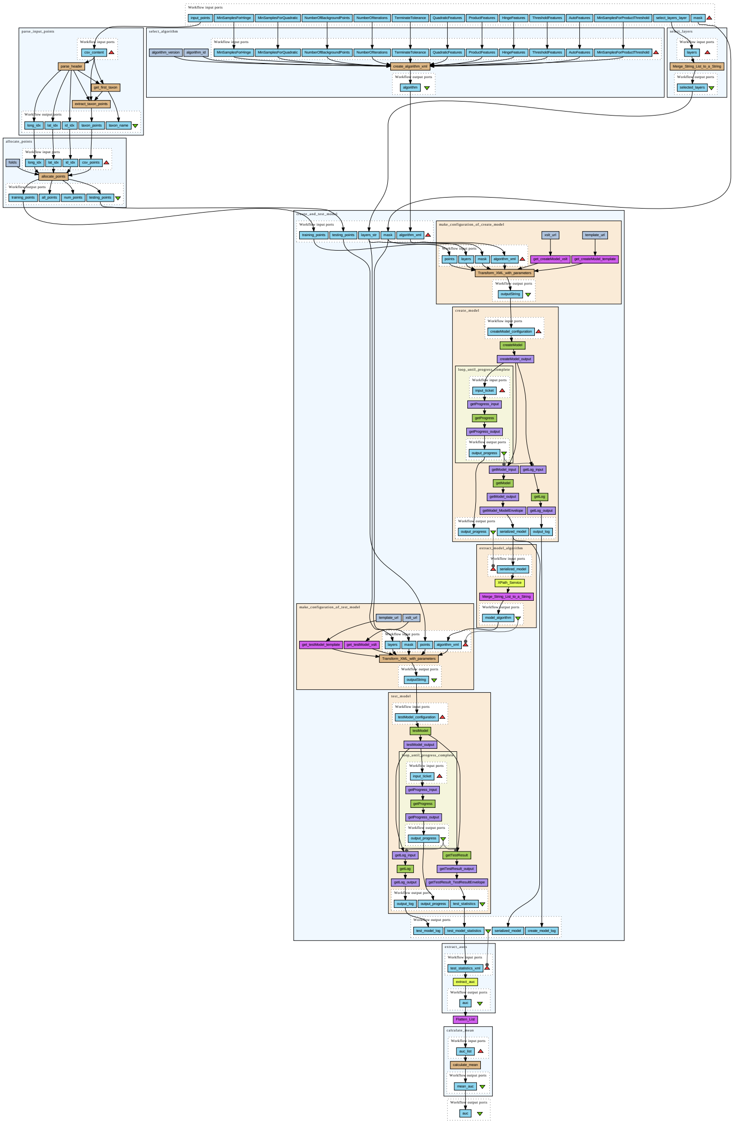


Figure A.5: ENM workflow for optimizing the maximum-entropy algorithm (Maxent). The workflow is available at myExperiment: http://www.myexperiment.org/workflows/3679.html.

A.2.4 *Crassostrea gigas*

A.2.5 Results of the SVM algorithm

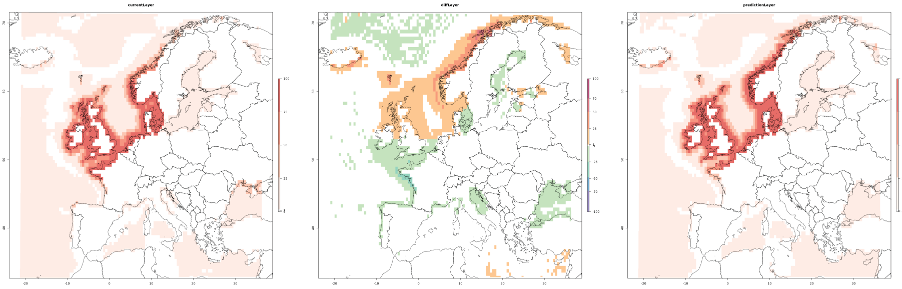
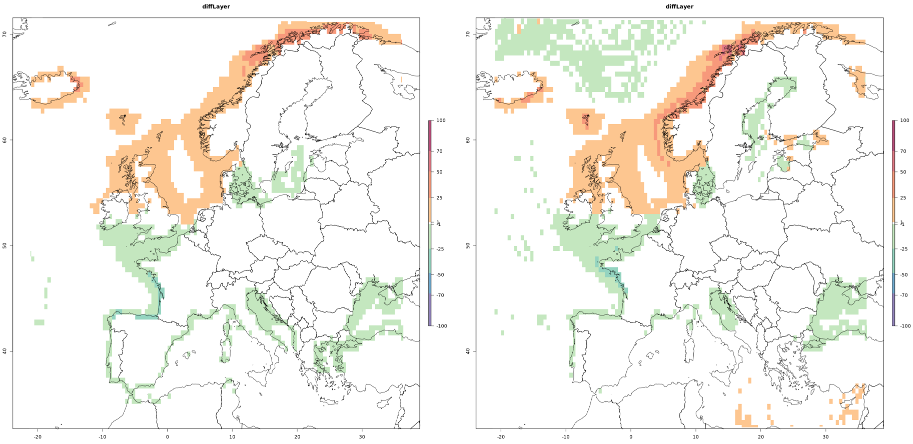


Figure A.6: *Crassostrea gigas*. Prediction of changes in habitat distribution today (left hand) until 2050 (right hand) as obtained from the SVM algorithm. The middle picture shows the difference of the two projections. "The habitat of the Pacific oyster will migrate northward in the coming decades. Typical oyster banks in the south (Brittany, Normandy) will become unsuitable, while northern locations (West coast of Norway and Northern Ireland and Britain) will gain suitability for this species. This has important implications for aquaculture industry. Some of the optimized models give more accurate projections of species distributions. Without the optimization, ENM workflow analyses rely on default parameter values or parameter free algorithms such as the Mahalonobis algorithm. In both cases, resulting models may be not sufficiently accurate." [Obst2013]



(a) The default SVM projection

(b) The optimized SVM projection

Figure A.7: The figures show the diffLayers of the default and optimized model to project the habitat distribution today and 2050.

A.2.6 *Prorocentrum minimum*

A.2.7 Results of the SVM algorithm

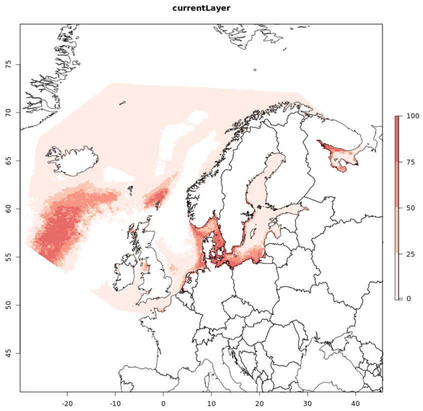
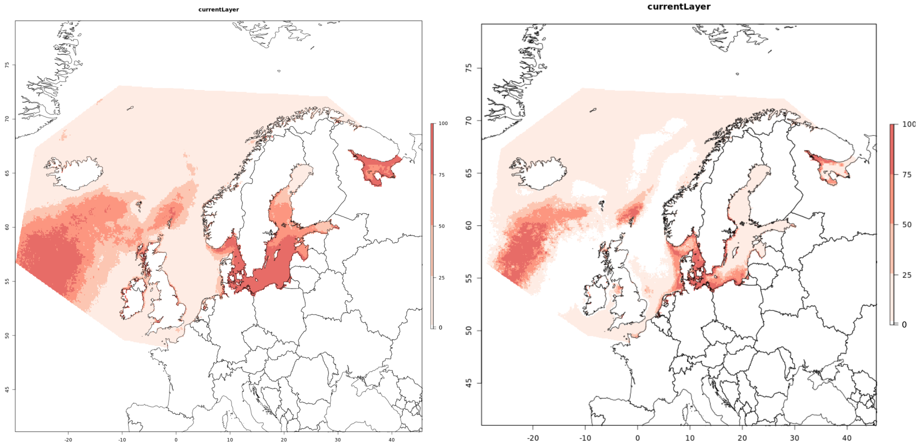


Figure A.8: *Prorocentrum minimum*: Prediction of current habitat distribution as obtained from the SVM algorithm. "*Prorocentrum minimum* is not likely to establish algal blooms in the central and Eastern Baltic. The main risk zone for Prorocentrum HABs are in the area of Kattegat, Skagerrak, and the Western Baltic." [Obst2013]



(a) The default SVM projection

(b) The optimized SVM projection

Figure A.9: The figures show the default (left hand) and optimized (right hand) model to project the current habitat distribution as obtained from the SVM algorithm.

A.3 The Research Object Optimization Ontology

A.3.1 Algorithm

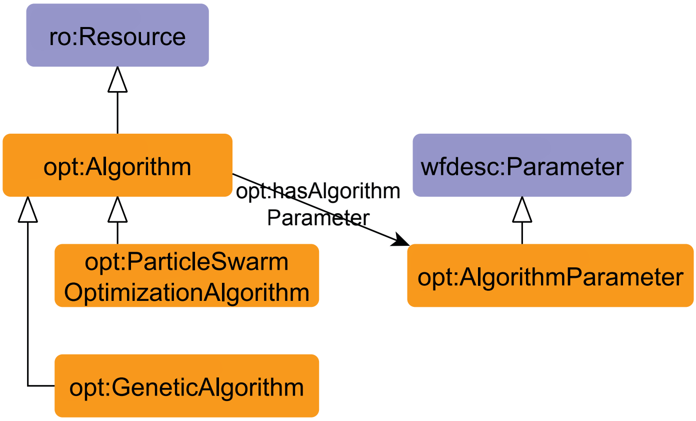


Figure A.10: "Classes and properties of the Algorithm section of RO-Opt. Blue concepts show the terms reused from the Research Object Ontologies." (Source: [Holl2013c].)

A.3.2 Fitness

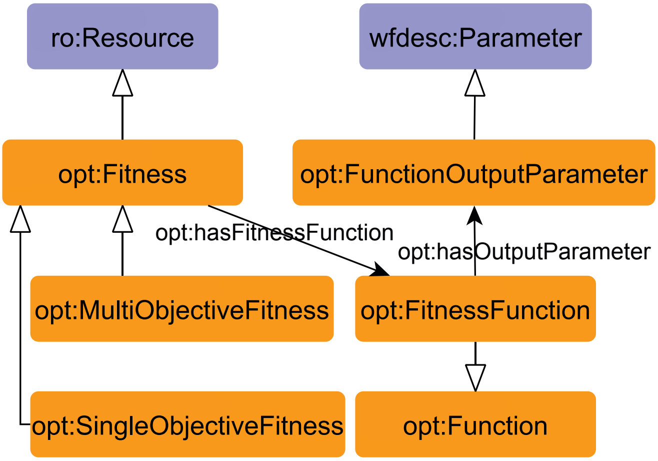


Figure A.11: "Classes and properties of the Fitness section of RO-Opt. Blue concepts show the terms reused from the Research Object Ontologies." (Source: [Holl2013c].)

A.3.3 Optimization Run

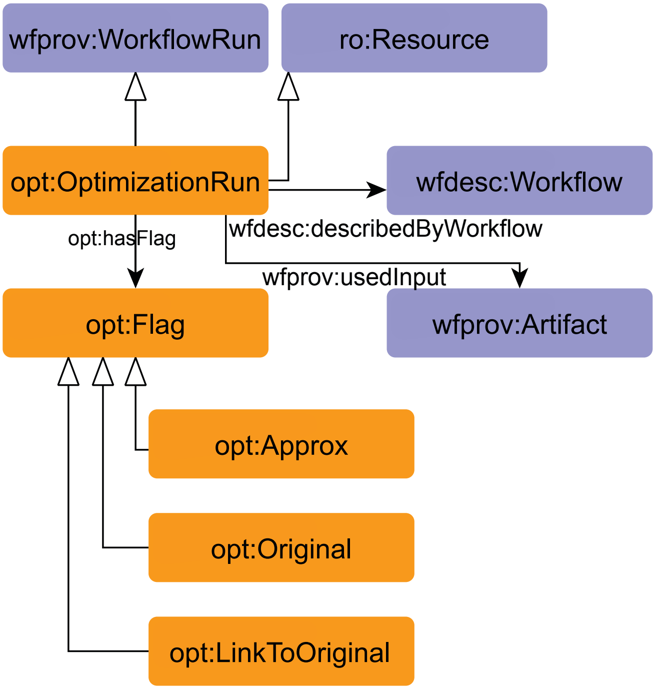


Figure A.12: "Classes and properties of the Optimization Run section of RO-Opt. Blue concepts show the terms reused from the Research Object Ontologies." (Source: [Holl2013c])

A.3.4 Search Space

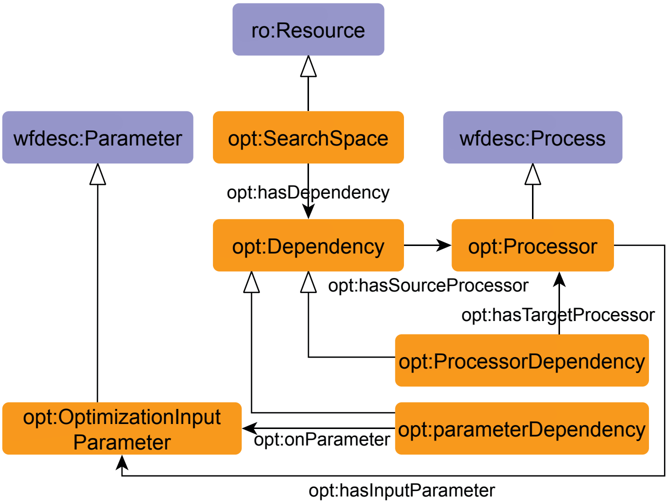


Figure A.13: "Classes and properties of the Search Space section of RO-Opt. The specific types of input parameters (numeric, string, etc.) and data properties have been omitted for simplicity. Blue concepts show the terms reused from the Research Object Ontologies." (Source: [Holl2013c].)

A.3.5 Optimization Provenance Ontology

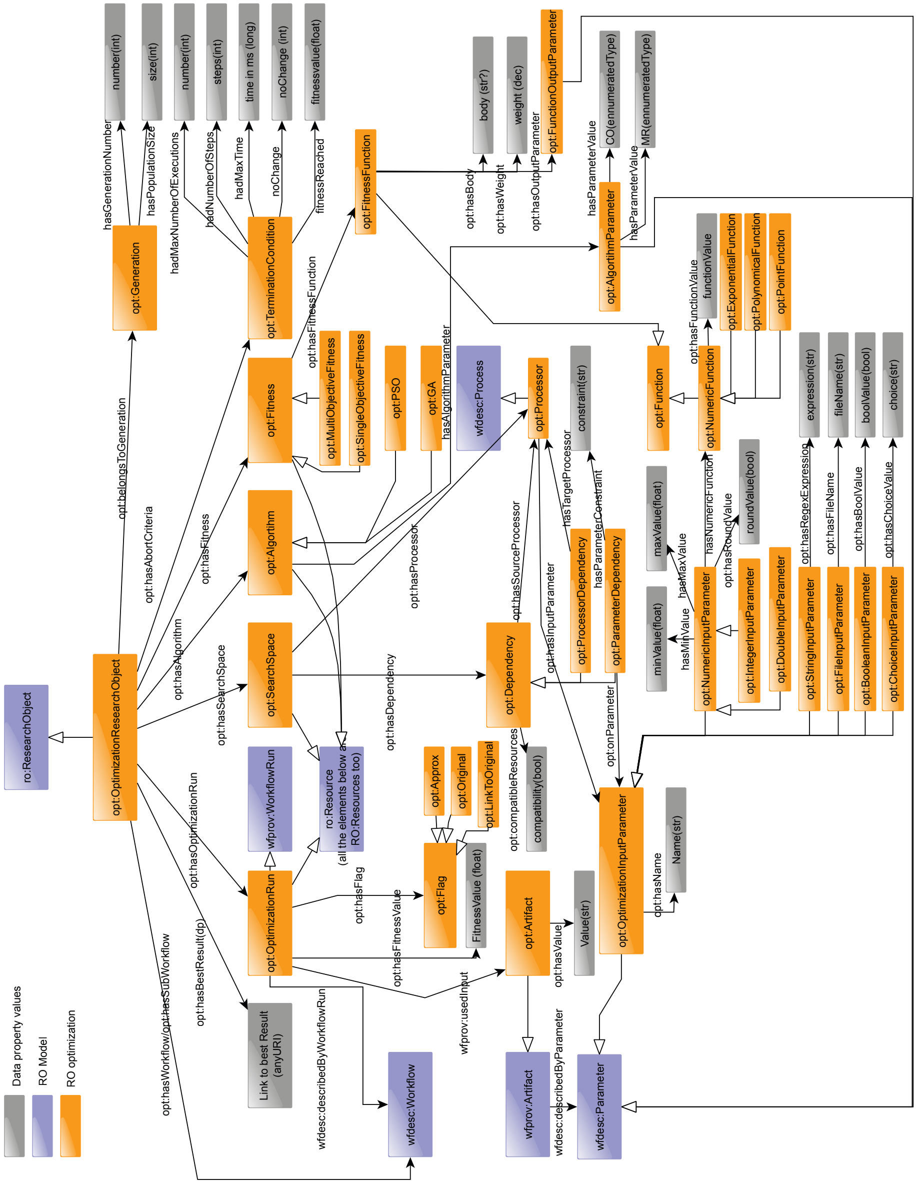


Figure A.14: Diagram showing all entities required to store optimization provenance data for reusage. (Source: [Holl2013b].)

A.3.6 BioVel Optimization Provenance

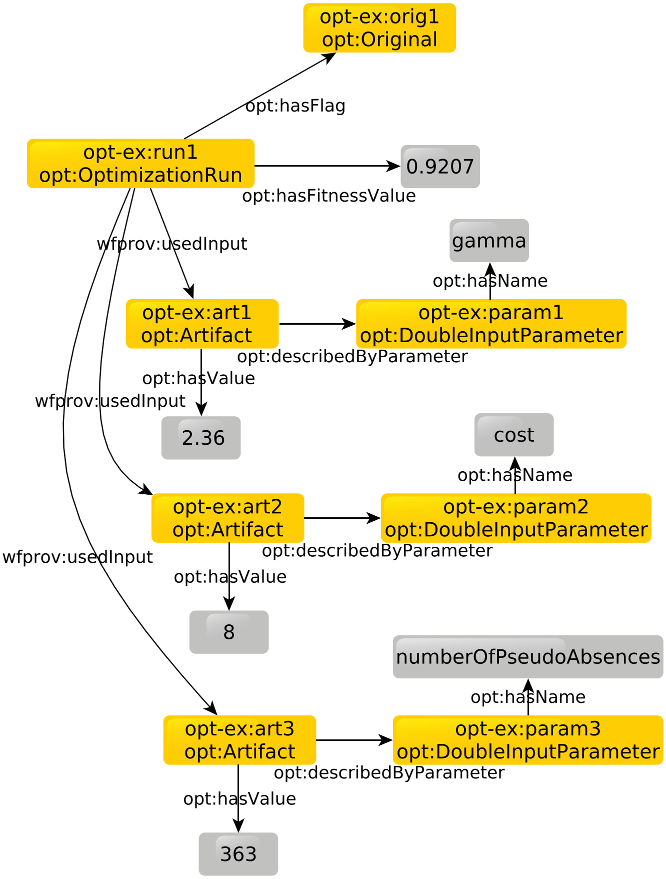


Figure A.15: "Modeling an Optimization run with the RO-Opt ontology (opt-ex:http://purl.org/net/svm-opt-research-object)." (Source: [Holl2013c].)

Bibliography

[Aalst2003] Wil van der Aalst, Arthur Ter Hofstede, Bartek Kiepuszewski, and Alistair P. Barros, „Workflow patterns“, *Distributed and parallel databases* 14 (1), 2003, pages 5–51.

[Aalst2004] Wil van der Aalst and Kees Max Van Hee, *Workflow Management: Models, Methods, and Systems*, The MIT press, 2004.

[Abeel2010] Thomas Abeel, Thibault Helleputte, Yves Van de Peer, Pierre Dupont, and Yvan Saeys, „Robust biomarker identification for cancer diagnosis with ensemble feature selection methods“, *Bioinformatics* 26 (3), 2010, pages 392–398.

[Abouelhoda2010] Mohamed Abouelhoda, Shadi A. Issa, and Moustafa Ghanem, „Meta-workflows: pattern-based interoperability between Galaxy and Taverna“, *Proceedings of the 1st International Workshop on Workflow Approaches to New Data-centric Science*, ACM, 2010, 2:1–2:8.

[Abouelhoda2012] Mohamed Abouelhoda, Shadi A. Issa, and Moustafa Ghanem, „Tavaxy: Integrating Taverna and Galaxy workflows with cloud computing support“, *BMC Bioinformatics* 13 (77), 2012.

[Abramson2009] David Abramson, Blair Bethwaite, Colin Enticott, Slavisa Garic, and Tom Peachey, „Parameter Space Exploration Using Scientific Workflows“, *Proceedings of the 9th International Conference on Computational Science*, vol. 5544, Lecture Notes in Computer Science, Springer, 2009, pages 104–113.

[Abramson2010] David Abramson, Blair Bethwaite, Colin Enticott, Slavisa Garic, Tom Peachey, Anushka Michailova, and Saleh Amirriazi, „Embedding optimization in computational science workflows“, *Journal of Computational Science* 1 (1), 2010, pages 41–47.

[Accelrys2007] Accelrys Software, Inc., *Pipeline Pilot is Accelrys’ scientific informatics platform*, 2007, URL: http://aaccelrys.com/pipeline-pilot (Accessed: 07/13/2013).

[Achilleos2012] K. G. Achilleos, C. C. Kannas, C. A. Nicolaou, C. S. Pattichis, and V. J. Promponas, „Open source workflow systems in life sciences informatics“, *Proceedings of the 12th International Conference on Bioinformatics & Bioengineering*, IEEE Computer Society, 2012, pages 552–558.

[Aebersold2003] Ruedi Aebersold and Matthias Mann, „Mass spectrometry-based proteomics“, *Nature* 422 (6928), 2003, pages 198–207.

[Alba2013] Enrique Alba, Gabriel Luque, and Sergio Nesmachnow, „Parallel metaheuristics: recent advances and new trends“, *International Transactions in Operational Research* 20 (1), 2013, pages 1–48.

[Alexander2011] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao, „Describing Linked Datasets with the VoID Vocabulary“, 2011, URL: http://www.w3.org/TR/void/ (Accessed: 07/05/2013).

[Allen2003] Gabrielle Allen, Tom Goodale, Thomas Radke, Michael Russell, Ed Seidel, Kelly Davis, Konstantinos N. Dolkas, Nikolaos D. Doulamis, Thilo Kielmann, André Merzky, Jarek Nabrzyski, Juliusz Pukacki, John Shalf, and Ian Taylor, „Enabling Applications on the Grid: A GridLab Overview“, *International Journal of High Performance Computing Applications* 17 (4), 2003, pages 449–466.

[AlRashidi2009] Mohammed R. AlRashidi and Mohamed E. El-Hawary, „A Survey of Particle Swarm Optimization Applications in Electric Power Systems“, *IEEE Transactions on Evolutionary Computation* 13 (4), 2009, pages 913–918.

[Altschul1990] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman, „Basic local alignment search tool“, *Journal of Molecular Biology* 215 (3), 1990, pages 403–410.

[Altschul1997] Stephen F. Altschul, Thomas L. Madden, Alejandro A. Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, and David J. Lipman, „Gapped BLAST and PSI-BLAST: a new generation of protein database search programs“, *Nucleic acids research* 25 (17), 1997, pages 3389–3402.

[Back1996] Thomas Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, 1996.

[Back2001] Thomas Bäck, „Introduction to the Special Issue: Self-Adaptation“, *Evolutionary Computation* 9 (2), 2001, pages 3–4.

[Bader2004] David A. Bader, „Computational Biology and High-Performance Computing“, *Communications of the ACM* 47 (11), 2004, pages 34–41.

[Bagewadi2013] Shweta Bagewadi, *Fraunhofer Institute for Algorithms and Scientific Computing (SCAI), Sankt Augustin, Germany*, Personal Communication, 2013.

[Baker2005] Mark Baker, Amy Apon, Clayton Ferner, and Jeff Brown, „Emerging Grid Standards“, *Computer* 38 (4), 2005, pages 43–50.

[Baker2010] Syed Murtuza Baker, Kai Schallau, and Björn H. Junker, „Comparison of different algorithms for simultaneous estimation of multiple parameters in kinetic metabolic models“, *Journal of Integrative Bioinformatics* 7 (3), 2010.

[Balasubram2005] R. Balasubramanian, S. Babak, D. Churches, and T. Cokelaer, „GEO600 Online Detector Characterization System“, *Classical and Quantum Gravity* 22 (23), 2005, page 4973.

[Barga2007] Roger Barga and Dennis Gannon, „Business Versus Scientific Workflow“, *Workflows for e-Science*, Springer, 2007, pages 9–16.

[Barker2008] Adam Barker and Jano Van Hemert, „Scientific Workflow: A Survey and Research Directions“, *Parallel Processing and Applied Mathematics*, vol. 4967, Lecture Notes in Computer Science, Springer, 2008, pages 746–753.

[BBSRC2013] BBSRC and BMBF, *JERM*, 2013, URL: http://www.sysmo-db.org/jerm (Accessed: 08/11/2013).

[Becanovic2010] Kristina Becanovic, Mahmoud A. Pouladi, Raymond S. Lim, Alexandre Kuhn, Paul Pavlidis, Ruth Luthi-Carter, Michael R. Hayden, and Blair R. Leavitt, „Transcriptional changes in Huntington disease identified using genome-wide expression profiling and cross-platform analysis“, *Human Molecular Genetics* 19 (8), 2010, pages 1438–1452.

[Bechhofer2013] Sean Bechhofer, Iain Buchan, David De Roure, Paolo Missier, John Ainsworth, Jiten Bhagat, Philip Couch, Don Cruickshank, Mark Delderfield, Ian Dunlop, Matthew Gamble, Carole Goble, and Danius Michaelides, „Why Linked Data is Not Enough for Scientists“, *Future Generation Computer Systems* 29 (2), 2013, pages 599–611.

[Belhajjame2012a] Khalid Belhajjame, Oscar Corcho, Daniel Garijo, Jun Zhao, Paolo Missier, David R. Newman, Raul Palma, Sean Bechhofer, Garcia Cuesta Esteban, Jose Manuel Gomez-Perez, Graham Klyne, Kevin Page, Marco Roos, José Enrique Ruiz, Stian Soiland-Reyes, Lourdes Verdes-Montenegro, David De Roure, and Carole Goble, „Workflow-Centric Research Objects: A First Class Citizen in the Scholarly Discourse“, *Proceedings of the ESWC 2012 Workshop on the Future of Scholarly Communication in the Semantic Web*, 2012, pages 1–12.

[Belhajjame2012b] Khalid Belhajjame, Daniel Garijo, Oscar Corcho, Esteban Garcia-Cuesta, and Stian Soiland-Reyes, *Wf4Ever Research Object Ontologies and Vocabularies Primer*, 2012, URL: http://wf4ever.github.io/ro-primer/ (Accessed: 08/23/2013).

[Belhajjame2013] Khalid Belhajjame, Jun Zhao, Daniel Garijo, Aleix Garrido, Stian Soiland-Reyes, Pinar Alper, and Oscar Corcho, „A Workflow PROV-Corpus based on Taverna and Wings“, *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, ACM, 2013, pages 331–332.

[Bell2009] Gordon Bell, Tony Hey, and Alex Szalay, „Beyond the Data Deluge“, *Science* 323 (5919), 2009, pages 1297–1298.

[Benedyczak2011] Krzysztof Benedyczak, Piotr Bała, Sven van den Berghe, Roger Menday, and Bernd Schuller, „Key aspects of the UNICORE 6 security model“, *Future Generation Computer Systems* 27 (2), 2011, pages 195–201.

[Berthold2008] Michael R. Berthold, Nicolas Cebron, Fabian Dill, Thomas R. Gabriel, Tobias Kötter, Thorsten Meinl, Peter Ohl, Christoph Sieb, Kilian Thiel, and Bernd Wiswedel, „KNIME: The Konstanz information miner“, *Data Analysis, Machine Learning and Applications*, Springer, 2008.

[Beyer2001] Hans-Georg Beyer and Kalyanmoy Deb, „On self-adaptive features in real-parameter evolutionary algorithms“, *IEEE Transactions on Evolutionary Computation* 5 (3), 2001, pages 250–270.

[Bhagat2010] Jiten Bhagat, Franck Tanoh, Eric Nzuobontane, Thomas Laurent, Jerzy Orlowski, Marco Roos, Katy Wolstencroft, Sergejs Aleksejevs, Robert Stevens, Steve Pettifer, Rodrigo Lopez, and Carole Goble, „BioCatalogue: a universal catalogue of web services for the life sciences“, *Nucleic acids research* 38 (suppl 2), 2010, pages 689–694.

[BioVel2012a] BioVel, *BioVeL Biodiversity Virtual e-Laboratory*, 2012, URL: https://wiki.biovel.eu/display/doc/Ecological+Niche+Modelling+Workflow (Accessed: 06/22/2013).

[BioVel2012b] BioVel, *BioVeL Biodiversity Virtual e-Laboratory*, 2012, URL: http://www.biovel.eu/ (Accessed: 06/22/2013).

[Bizer2009] Christian Bizer, Tom Heath, and Tim Berners-Lee, „Linked Data - The Story So Far“, *International Journal on Semantic Web and Information Systems* 5 (3), 2009, pages 1–22.

[Blum2011] Christian Blum, Jakob Puchinger, Günther R. Raidl, and Andrea Roli, „Hybrid metaheuristics in combinatorial optimization: A survey“, *Applied Soft Computing* 11 (6), 2011, pages 4135–4151.

[Boisson2008] Jean-Charles Boisson, Laetitia Jourdan, El-Ghazali Talbi, and Dragos Horvath, „Parallel multi-objective algorithms for the molecular docking problem“, *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, IEEE, 2008, pages 187–194.

[Branke2004] Jürgen Branke, Kalyanmoy Deb, Henning Dierolf, and Matthias Osswald, „Finding Knees in Multi-objective Optimization“, *Parallel Problem Solving from Nature-PPSN VIII*, Springer, 2004, pages 722–731.

[Brazma2001] Alvis Brazma, Pascal Hingamp, John Quackenbush, Gavin Sherlock, Paul Spellman, Chris Stoeckert, John Aach, Wilhelm Ansorge, Catherine A. Ball, Helen C. Causton, Terry Gaasterland, Patrick Glenisson, Frank C.P Holstege, Irene F. Kim, Victor Markowitz, John C. Matese, Helen Parkinson, Alan Robinson, Ugis Sarkans, Steffen Schulze-Kremer, Jason Stewart, Ronald Taylor, Jaak Vilo, and Martin Vingron, „Minimum information about a microarray experiment (MIAME) – toward standards for microarray data“, *Nature Genetics* 29 (4), 2001, pages 365–371.

[Cardiff2012] Cardiff University, *Triana - Open Source Problem Solving Software*, 2012, URL: http://www.trianacode.org (Accessed: 07/17/2013).

[Chang2011] Chih-Chung Chang and Chih-Jen Lin, „LIBSVM: a library for support vector machines“, *ACM Transactions on Intelligent Systems and Technology* 2 (3), 2011, 27:1–27:27.

[Chen2011] Li Chen, Jianhua Xuan, Rebecca Riggins, Robert Clarke, and Yue Wang, „Identifying cancer biomarkers by network-constrained support vector machines“, *BMC Systems Biology* 5 (161), 2011.

[Chen2009] Wei-Neng Chen and Jun Zhang, „An Ant Colony Optimization Approach to a Grid Workflow Scheduling Problem With Various QoS Requirements“, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 39 (1), 2009, pages 29–43.

[Chinthaka2009] Eran Chinthaka, Jaliya Ekanayake, David Leake, and Beth Plale, „CBR Based Workflow Composition Assistant“, *2009 World Conference on Services-I*, IEEE, 2009, pages 352–355.

[Chirigati2012] Fernando Chirigati, Vítor Silva, Eduardo Ogasawara, Daniel de Oliveira, Jonas Dias, Fábio Porto, Patrick Valduriez, and Marta Mattoso, „Evaluating Parameter Sweep Workflows in High Performance Computing“, *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*, SWEET ’12, ACM, 2012, 2:1–2:10.

[Chung2006] I-Hsin Chung and Jeffrey K. Hollingsworth, „A Case Study Using Automatic Performance Tuning for Large-Scale Scientific Programs“, *Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing*, IEEE, 2006, pages 45–56.

[Cobb1993] Helen G. Cobb and John J. Grefenstette, „Genetic Algorithms for Tracking Changing Environments“, *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann, 1993, pages 523–530.

[Coello2002] Carlos A. Coello Coello, „Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art“, *Computer methods in applied mechanics and engineering* 191 (11), 2002, pages 1245–1287.

[CohenBoula2011] Sarah Cohen-Boulakia and Ulf Leser, „Search, adapt, and reuse: the future of scientific workflows“, *ACM SIGMOD Record* 40 (2), 2011, pages 6–16.

[Colaco2009] Marcelo J. Colaço and George S. Dulikravich, „A Survey of Basic Deterministic, Heuristic and Hybrid Methods for Single Objective Optimization and Response Surface Generation“, *Thermal Measurements and Inverse Techniques*, 2009, pages 355–405.

[Craig2004] Robertson Craig and Ronald C. Beavis, „TANDEM: matching proteins with tandem mass spectra“, *Bioinformatics* 20 (9), 2004, pages 1466–1467.

[Crick2009] Tom Crick, Peter Dunning, Hyunsun Kim, and Julian Padget, „Engineering design optimization using services and workflows“, *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 367 (1898), 2009, pages 2741–2751.

[Curcin2008] Vasa Curcin and Moustafa Ghanem, „Scientific workflow systems-can one size fit all?“, *Cairo International Biomedical Engineering Conference*, IEEE, 2008, pages 1–9.

[Davis1989] Lawrence Davis, „Adapting Operator Probabilities in Genetic Algorithms“, *Proceedings of the third international conference on Genetic algorithms*, Morgan Kaufmann, 1989, pages 61–69.

[De Brevern2000] Alexandre G. De Brevern, Catherine Etchebest, and Serge Hazout, „Bayesian probabilistic approach for predicting backbone structures in terms of protein blocks“, *Proteins: Structure, Function, and Bioinformatics* 41 (3), 2000, pages 271–287.

[De Roure2009a] David De Roure and Carole Goble, „Lessons from myExperiment: Research Objects for Data Intensive Research“, *Microsoft e-Science Workshop*, 2009.

[De Roure2009b] David De Roure and Carole Goble, „myExperiment: A Web 2.0 Virtual Research Environment for Research using Computation and Services“, *Workshop On Integrating Digital Library Content with Computational Tools and Services*, 2009.

[De Roure2009c] David De Roure and Carole Goble, „Software Design for Empowering Scientists“, *Software, IEEE* 26 (1), 2009, pages 88–95.

[De Roure2010] David De Roure, Carole Goble, Sergejs Aleksejevs, Sean Bechhofer, Jiten Bhagat, Don Cruickshank, Paul Fisher, Nandkumar Kollara, Danius Michaelides, Paolo Missier, David Newman, Marcus Ramsden, Marco Roos, Katy Wolstencroft, Ed Zaluska, and Jun Zhao, „The Evolution of myExperiment“, *2010 IEEE Sixth International Conference on e-Science (e-Science)*, IEEE, 2010, pages 153–160.

[De Roure2009d] David De Roure, Carole Goble, and Robert Stevens, „The Design and Realisation of the myExperiment Virtual Research Environment for Social Sharing of Workflows“, *Future Generation Computer Systems* 25 (5), 2009, pages 561–567.

[Deb2001] Kalyanmoy Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, 2001.

[Deb2002a] Kalyanmoy Deb, Ashish Anand, and Dhiraj Joshi, „A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization“, *Evolutionary computation* 10 (4), 2002, pages 371–395.

[Deb2002b] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and Tamt Meyarivan, „A fast and elitist multiobjective genetic algorithm: NSGA-II“, *IEEE Transactions on Evolutionary Computation* 6 (2), 2002, pages 182–197.

[Debye1909] Peter Debye, „Näherungsformeln für die Zylinderfunktionen für große Werte des Arguments und unbeschränkt veränderliche Werte des Index“, *Mathematische Annalen* 67 (4), 1909, pages 535–558.

[Deelman2009] Ewa Deelman, Dennis Gannon, Matthew Shields, and Ian Taylor, „Workflows and e-Science: An overview of workflow system features and capabilities“, *Future Generation Computer Systems* 25 (5), 2009, pages 528–540.

[Deelman2006a] Ewa Deelman and Yolanda Gil, „Managing Large-Scale Scientific Workflows in Distributed Environments: Experiences and Challenges“, *2006 IEEE Second International Conference on e-Science and Grid Computing (e-Science)*, IEEE, 2006, pages 144–149.

[Deelman2006b] Ewa Deelman, Yolanda Gil, and Maria Zemankova, „NSF Workshop on the Challenges of Scientific Workflows“, *The National Science Foundation’s Workshop*, 2006, pages 1–2.

[Deelman2005] Ewa Deelman, Gurmeet Singh, Mei-Hui Su, James Blythe, Yolanda Gil, Carl Kesselman, Gaurang Mehta, Karan Vahi, G Bruce Berriman, John Good, Anastasia Laity, Joseph C. Jacob, and Daniel S. Katz, „Pegasus: A framework for mapping complex scientific workflows onto distributed systems“, *Scientific Programming* 13 (3), 2005, pages 219–237.

[DEISA2013] DEISA, *DEISA - Distributed European Infrastructure for Supercomputing Applications*, 2013, URL: http://www.deisa.org (Accessed: 05/13/2013).

[Demuth2010] Bastian Demuth, Sonja Holl, and Bernd Schuller, „Extended Execution Support for Scientific Applications in Grid Environments", *Proceedings of UNICORE Summit 2010*, Schriften des Forschungszentrums Jülich, IAS Series 5, Forschungszentrum Jülich, 2010, pages 61–67.

[D-Grid2013] D-Grid, *German National Grid: D-Grid*, 2013, URL: http://www.d-grid.de (Accessed: 05/13/2013).

[Di Nuovo2012] Alessandro G. Di Nuovo, Giuseppe Ascia, and Vincenzo Catania, „A Study on Evolutionary Multi-Objective Optimization with Fuzzy Approximation for Computational Expensive Problems", *Parallel Problem Solving from Nature*, vol. 7492, Lecture Notes in Computer Science, Springer, 2012, pages 102–111.

[DiazGomez2007] Pedro A. Diaz-Gomez and Dean F. Hougen, „Initial Population for Genetic Algorithms: A Metric Approach", *Proceedings of the 2007 International Conference on Genetic and Evolutionary Methods*, 2007, pages 43–49.

[Dorigo1999] Marco Dorigo and Gianni Di Caro, „Ant Colony Optimization: A New Meta-Heuristic", *Proceedings of the 1999 Congress on Evolutionary Computation*, vol. 2, IEEE, 1999.

[Drescher2009] Michael Drescher, Ali Anjomshoaa, Geoff Williams, and David Meredith, *JSDL Parameter Sweep Job Extension*, 2009, URL: http://www.ogf.org/documents/GFD.149.pdf (Accessed: 09/15/2013).

[Dyer2010] Daniel W. Dyer, *Watchmaker Framework for Evolutionary Computation*, 2010, URL: http://watchmaker.uncommons.org/ (Accessed: 05/13/2013).

[EBI2010] EBI, *Minimum Information About a Microarray Experiment - MIAME*, 2010, URL: http://www.mged.org/Workgroups/MIAME/miame.html (Accessed: 08/07/2013).

[Edgar2002] Ron Edgar, Michael Domrachev, and Alex E. Lash, „Gene Expression Omnibus: NCBI gene expression and hybridization array data repository“, *Nucleic Acids Research* 30 (1), 2002, pages 207–210.

[Engelbrech2005] Andries P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, vol. 1, Wiley, 2005.

[Enticott2010] Colin Enticott, Thomas Peachey, David Abramson, Elena Mashkina, Chong-Yong Lee, Alan Bond, Gareth Kennedy, David Gavaghan, and Darrell Elton, „Electrochemical Parameter Optimization Using Scientific Workflows“, *2010 IEEE Sixth International Conference on e-Science (e-Science)*, IEEE, 2010, pages 324–330.

[Eshelman1993] Larry J. Eshelman and J. David Schaffer, „Real-Coded Genetic Algorithms and Interval-Schemata“, *Foundation of Genetic Algorithms 2*, ed. by Darrell L. Whitley, Morgan Kaufmann, 1993, pages 187–202.

[Fan2011] Xiaoliang Fan, Patrick Brézillon, Ruisheng Zhang, and Lian Li, „Making context explicit towards decision support for a flexible scientific workflow system“, *Fourth Workshop on Human Centered Processes*, 2011, pages 3–9.

[Floudas2007] Christodoulos Achilleus Floudas, „Computational methods in protein structure prediction“, *Biotechnology and bioengineering* 97 (2), 2007, pages 207–213.

[Foster2005] Ian Foster, „Globus Toolkit Version 4: Software for Service-Oriented Systems“, *Network and parallel computing*, vol. 3779, Lecture Notes in Computer Science, Springer, 2005, pages 2–13.

[Fox2006] „Special issue: Workflow in grid systems“, *Concurrency and Computation: Practice and Experience* 18 (10), 2006, ed. by Geoffrey C. Fox and Dennis Gannon, pages 1009–1019.

[Fraunhofer2013] Fraunhofer SCAI, *AETIONOMY: About Aetionomy*, 2013, URL: http://purl.org/net/svm-opt-research-object (Accessed: 09/15/2013).

[Frottin2006] Frédéric Frottin, Aude Martinez, Philippe Peynot, Sanghamitra Mitra, Richard C. Holz, Carmela Giglione, and Thierry Meinnel, „The proteomics of N-terminal methionine cleavage“, *Molecular & Cellular Proteomics* 5 (12), 2006, pages 2336–2349.

[Galaxy2013] Galaxy, *The Galaxy Project: Online bioinformatics analysis for everyone*, 2013, URL: http://galaxyproject.org (Accessed: 07/13/2013).

[Gamble2012] Matthew Gamble, Carole Goble, Graham Klyne, and Jun Zhao, „MIM: A Minimum Information Model vocabulary and framework for Scientific Linked Data“, *2012 IEEE 8th International Conference on E-Science (e-Science)*, IEEE, 2012, pages 1–8.

[GBIF2009] Botanical Garden and Botanical Museum Berlin-Dahlem, *Global Biodiversity Information Facility*, 2009, URL: http://www.gbif.org (Accessed: 08/14/2013).

[Geer2010] Lewis Y. Geer, Aron Marchler-Bauer, Renata C. Geer, Lianyi Han, Jane He, Siqian He, Chunlei Liu, Wenyao Shi, and Stephen H. Bryant, „The NCBI BioSystems database“, *Nucleic Acids Research* 38 (suppl 1), 2010, pages D492–D496.

[Gendreau2010] Michel Gendreau and Jean-Yves Potvin, eds., *Handbook of Metaheuristics*, 2nd edition, Springer, 2010.

[Gibson2013] R. J. Gibson, A. I. Nepomuceno, S. M. Randall, N. Muthusamy, H. T. Ghashghaei, and D. C. Muddiman, *Elucidation of Search Parameters for Q-Exactive to Maximize Protein Identifications at 1% False Discovery Rate Using Wild-Type and FoxJ1 Knock Out Mouse Brain Tissues*, Poster presented at the 61st ASMS Conference on Mass Spectrometry and Allied Topics, 2013.

[Gil2008] Yolanda Gil, „From Data to Knowledge to Discoveries: Scientific Workflows and Artificial Intelligence“, *Scientific Programming* 16 (4), 2008.

[Gil2007] Yolanda Gil, Ewa Deelman, Mark Ellisman, Thomas Fahringer, Geoffrey Fox, Dennis Gannon, Carole Goble, Miron Livny, Luc Moreau, and Jim Myers, „Examining the Challenges of Scientific Workflows“, *Computer* 40 (12), 2007, pages 24–32.

[Gil2011] Yolanda Gil, Varun Ratnakar, Jihie Kim, Joshua Moody, Ewa Deelman, Pedro Antonio González-Calero, and Paul Groth, „Wings: Intelligent Workflow-Based Design of Computational Experiments“, *Intelligent Systems, IEEE* 26 (1), 2011, pages 62–72.

[Giovanni2013] Renato De Giovanni, *Reference Center on Environmental Information, Campinas SP, Brazil*, Personal Communication, 2013.

[Goble2010] Carole Goble, Jiten Bhagat, Sergejs Aleksejevs, Don Cruickshank, Danius Michaelides, David Newman, Mark Borkum, Sean Bechhofer, Marco Roos, Peter Li, and David De Roure, „myExperiment: a repository and social network for the sharing of bioinformatics workflows“, *Nucleic acids research* 38 (suppl 2), 2010, W677–W682.

[Goble2007] Carole Goble and David De Roure, „myExperiment: social networking for workflow-using e-scientists“, *Proceedings of the 2nd Workshop on Workflows in Support of Large-Scale Science*, ACM, 2007, pages 1–2.

[Goble2009] Carole Goble, Paolo Missier, and David De Roure, „Scientific Workflows“, *McGraw-Hill Yearbook of Science & Technology 2009*, McGraw-Hill, 2009.

[Goecks2010] Jeremy Goecks, Anton Nekrutenko, James Taylor, and The Galaxy Team, „Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences“, *Genome Biology* 11 (8), 2010, R86.

[Goldberg1990] David E. Goldberg, „Real-coded Genetic Algorithms, Virtual Alphabets, and Blocking“, *Complex Systems* 5, 1990, pages 139–167.

[Goldberg1988] David E. Goldberg and John H. Holland, „Genetic algorithms and machine learning“, *Machine learning* 3 (2), 1988, pages 95–99.

[Goldstein1971] A.A. Goldstein and J.F. Price, „On descent from local minima“, *Mathematics of Computation* 25 (115), 1971, pages 569–574.

[Gondro2007] Cedric Gondro and Brian P. Kinghorn, „A simple genetic algorithm for multiple sequence alignment“, *Genetics and Molecular Research* 6 (4), 2007, pages 964–982.

[Gorlach2011] Katharina Görlach, Mirko Sonntag, Dimka Karastoyanova, Frank Leymann, and Michael Reiter, „Conventional Workflow Technology for Scientific Simulation“, *Guide to e-Science*, Computer Communications and Networks, Springer, 2011, pages 323–352.

[Gotoh1982] Osamu Gotoh, „An improved algorithm for matching biological sequences“, *Journal of Molecular Biology* 162 (3), 1982, pages 705–708.

[Guinan2009] Janine Guinan, Colin Brown, Margaret F.J. Dolan, and Anthony J. Grehan, „Ecological niche modelling of the distribution of cold-water coral habitat using underwater remote sensing data“, *Ecological Informatics* 4 (2), 2009, pages 83–92.

[Hadka2012] Dave Hadka et al., *MOEA Framework*, 2012, URL: http://www.moeaframework.org/ (Accessed: 06/27/2013).

[Handl2007] Julia Handl, Douglas B. Kell, and Joshua Knowles, „Multiobjective optimization in bioinformatics and computational biology“, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 4 (2), 2007, pages 279–292.

[Heath2012] Christopher Heath and Justin Gray, „OpenMDAO: Framework for Flexible Multidisciplinary Design, Analysis and Optimization Methods“, *8th AIAA Multidisciplinary Design Optimization Specialist Conference (MDO)*, 2012, pages 1–13.

[Herrera2003] Francisco Herrera, Manuel Lozano, and Ana M Sánchez, „A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study“, *International Journal of Intelligent Systems* 18 (3), 2003, pages 309–338.

[Herrera1998] Francisco Herrera, Manuel Lozano, and Jose L. Verdegay, „Tackling Real-Coded Genetic Algorithms: Operators and Tools for Behavioural Analysis“, *Artificial Intelligence Review* 12 (4), 1998, pages 265–319.

[Hey2009] Tony Hey, Stewart Tansley, and Kristin Tolle, eds., *The Fourth Paradigm: Data-Intensive Scientific Discovery*, Redmond, Washington: Microsoft Research, 2009.

[Hey2002] Tony Hey and Anne E. Trefethen, „The UK e-Science Core Programme and the Grid“, *Future Generation Computer Systems* 18 (8), 2002, pages 1017–1031.

[Hey2003] Tony Hey and Anne E. Trefethen, „The Data Deluge: An e-Science Perspective“, *Grid Computing - Making the Global Infrastructure a Reality*, Chapter: 36, Wiley and Sons, 2003, pages 809–824.

[Hey2005] Tony Hey and Anne E. Trefethen, „Cyberinfrastructure for e-Science“, *Science* 308 (5723), 2005, pages 817–821.

[Holl2013a] Sonja Holl, Daniel Garijo, and Khalid Belhajjame, *Parameter Optimization: BioVel Example using SVM*, 2013, URL: http://purl.org/net/svm-opt-research-object (Accessed: 08/23/2013).

[Holl2013b] Sonja Holl, Daniel Garijo, and Khalid Belhajjame, *RO-Opt Prefix*, 2013, URL: http://purl.org/net/RO-optimization# (Accessed: 08/23/2013).

[Holl2013c] Sonja Holl, Daniel Garijo, Khalid Belhajjame, Olav Zimmermann, Renato De Giovanni, Matthias Obst, and Carole Goble, „On Specifying and Sharing Scientific Workflow Optimization Results Using Research Objects“, *The 8th Workshop on Workflows in Support of Large-Scale Science*, to be published, IEEE, 2013.

[Holl2013d] Sonja Holl, Mohammad Shahbaz Memon, Morris Riedel, Yassene Mohammed, Magnus Palmblad, and Andrew Grimshaw, „Enhanced Resource Management enabling Standard Parameter Sweep Jobs for Scientific Applications“, *9th International Workshop on Scheduling and Resource Management for Parallel and Distributed Systems*, to be published, IEEE, 2013.

[Holl2013e] Sonja Holl, Yassene Mohammed, André M. Deelder, Olav Zimmermann, and Magnus Palmblad, „Optimized Scientific Workflows for Improved Peptide and Protein Identification“, *Molecular & Cellular Proteomics*, 2013, submitted: 4/7/2013.

[Holl2012a] Sonja Holl, Olav Zimmermann, Bastian Demuth, Bernd Schuller, and Martin Hofmann-Apitius, „Secure Multi-Level Parallel Execution of Scientific Workflows on HPC Grid Resources by Combining

Taverna and UNICORE Services“, *Proceedings of UNICORE Summit 2012*, Schriften des Forschungszentrums Jülich, IAS Series 15, Forschungszentrum Jülich, 2012, pages 27–34.

[Holl2011] Sonja Holl, Olav Zimmermann, and Martin Hofmann-Apitius, „A UNICORE Plugin for HPC-Enabled Scientific Workflows in Taverna 2.2“, *2011 IEEE World Congress on Services*, IEEE, 2011, pages 220–223.

[Holl2012b] Sonja Holl, Olav Zimmermann, and Martin Hofmann-Apitius, „A new optimization phase for scientific workflow management systems“, *2012 IEEE 8th International Conference on E-Science (e-Science)*, IEEE, 2012, pages 1–8.

[Holl2013f] Sonja Holl, Olav Zimmermann, Magnus Palmblad, Yassene Mohammed, and Martin Hofmann-Apitius, „A New Optimization Phase for Scientific Workflow Management Systems“, *Future Generation Computer Systems*, 2013, to be published.

[Holland1992a] John H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*, The MIT press, 1992.

[Holland1992b] John H. Holland, „Genetic algorithms“, *Scientific american* 267 (1), 1992, pages 66–72.

[Holtzer1954] Alfred M. Holtzer, „The collected papers of Peter J. W. Debye“, *Journal of Polymer Science* 13 (72), 1954, pages 548–548.

[Howe2008] Doug Howe, Maria Costanzo, Petra Fey, Takashi Gojobori, Linda Hannick, Winston Hide, David P. Hill, Renate Kania, Mary Schaeffer, Susan St Pierre, Simon Twigger, Owen White, and Seung Yon Rhee, „Big data: The future of biocuration“, *Nature* 455 (7209), 2008, pages 47–50.

[Hutchinson1957] G. Evelyn Hutchinson, „Cold Spring Harbor Symposium on Quantitative Biology“, *Concluding remarks* 22, 1957, pages 415–427.

[Ishibuchi1996] Hisao Ishibuchi and Tadahiko Murata, „Multi-objective genetic local search algorithm“, *Proceedings of IEEE International Conference on Evolutionary Computation*, IEEE, 1996, pages 119–124.

[Jagla2011] Bernd Jagla, Bernd Wiswedel, and Jean-Yves Coppée, „Extending KNIME for next-generation sequencing data analysis“, *Bioinformatics* 27 (20), 2011, pages 2907–2909.

[Jimenez2013] Rafael C. Jimenez and Manuel Corpas, „Bioinformatics Workflows and Web Services in Systems Biology Made Easy for Experimentalists“, *In Silico Systems Biology*, vol. 1021, Methods in Molecular Biology, Springer, 2013, pages 299–310.

[Jin2005] Yaochu Jin, „A Comprehensive Survey of Fitness Approximation in Evolutionary Computation“, *Soft computing* 9 (1), 2005, pages 3–12.

[Juve2012a] Gideon Juve, Ann Chervenak, Ewa Deelman, Shishir Bharathi, Gaurang Mehta, and Karan Vahi, „Characterizing and profiling scientific workflows“, *Future Generation Computer Systems* 29 (3), 2012, pages 682–692.

[Juve2012b] Gideon Juve, Ewa Deelman, G. Bruce Berriman, Benjamin P. Berman, and Philip Maechling, „An Evaluation of the Cost and Performance of Scientific Workflows on Amazon EC2“, *Journal of Grid Computing* 10 (1), 2012, pages 5–21.

[Kawas2006] Edward Kawas, Martin Senger, and Mark Wilkinson, „BioMoby extensions to the Taverna workflow management and enactment software“, *BMC Bioinformatics* 7 (523), 2006.

[Keller2005] Andrew Keller, Jimmy Eng, Ning Zhang, Xiao-jun Li, and Ruedi Aebersold, „A uniform proteomics MS/MS analysis platform utilizing open XML file formats“, *Molecular Systems Biology* 1 (0017), 2005.

[Keller2002] Andrew Keller, Alexey I. Nesvizhskii, Eugene Kolker, and Ruedi Aebersold, „Empirical Statistical Model To Estimate the Accuracy of Peptide Identifications Made by MS/MS and Database Search“, *Analytical chemistry* 74 (20), 2002, pages 5383–5392.

[Kennedy1995] James Kennedy and Russell Eberhart, „Particle swarm optimization“, *IEEE International Conference on Neural Networks*, vol. 4, 1995, pages 1942–1948.

[Kim2013] Sangtae Kim, Gordon W. Slysz, Kevin L. Crowell, Samuel H. Payne, Gordon A. Anderson, and Richard D. Smith, *IPA: an Informed Proteomics Analysis Tool for Improved Peptide Identifications*, Poster presented at the 61st ASMS Conference on Mass Spectrometry and Allied Topics, 2013.

[Kirkpatric1983] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi, „Optimization by simulated annealing“, *Science* 220 (4598), 1983, pages 671–680.

[KNIME2013] KNIME.com AG, *KNIME: Konstanz Information Miner*, 2013, URL: http://www.knime.org (Accessed: 07/13/2013).

[Koop2008] David Koop, Carlos E. Scheidegger, Steven P. Callahan, Juliana Freire, and Cláudio T. Silva, „VisComplete: Automating suggestions for visualization pipelines“, *IEEE Transactions on Visualization and Computer Graphics* 14 (6), 2008, pages 1691–1698.

[Krabbenhof2008] Hajo N. Krabbenhöft, Steffen Möller, and Daniel Bayer, „Integrating ARC grid middleware with Taverna workflows“, *Bioinformatics* 24 (9), 2008, pages 1221–1222.

[Krokhin2006] Oleg V. Krokhin, „Sequence-specific retention calculator. Algorithm for peptide retention prediction in ion-pair RP-HPLC: application to 300-and 100-Å pore size C18 sorbents“, *Analytical chemistry* 78 (22), 2006, pages 7785–7795.

[Kulhanek2011] Stefanie A. Kulhanek, Brian Leung, and Anthony Ricciardi, „Using ecological niche models to predict the abundance and impact of invasive species: application to the common carp“, *Ecological Applications* 21 (1), 2011, pages 203–213.

[Kulyk2006] Olga Kulyk and Ingo Wassink, „Getting to know bioinformaticians: Results of an exploratory user study“, *In Proceedings of British Computer Society Human Computer Interaction Conference*, 2006, pages 30–37.

[Kulyk2008] Olga Kulyk, Ingo Wassink, Paul van der Vet, Gerrit van der Veer, and Betsy van Dijk, *Sticks, balls or a ribbon? Results of a formative user study with bioinformaticians*, Dec. 2008.

[Kumar2010] Vijay S. Kumar, Tahsin Kurc, Varun Ratnakar, Jihie Kim, Gaurang Mehta, Karan Vahi, Yoonju Lee Nelson, P. Sadayappan, Ewa Deelman, Yolanda Gil, Mary Hall, and Joel Saltz, „Parameterized Specification, Configuration and Execution of Data-Intensive Scientific Workflows“, *Cluster computing* 13 (3), 2010, pages 315–333.

[Land1960] Ailsa H. Land and Alison G. Doig, „An Automatic Method of Solving Discrete Programming Problems“, *Econometrica: Journal of the Econometric Society* 28 (3), 1960, pages 497–520.

[Leake2008] David Leake and Joseph Kendall-Morwick, „Towards Case-Based Support for e-Science Workflow Generation by Mining Provenance“, *Advances in Case-Based Reasoning*, vol. 5239, Lecture Notes in Computer Science, Springer, 2008, pages 269–283.

[Lim2012] Yu Hua Lim, Jefferson Tana, and David Abramsonb, „Solving Optimization Problems in Nimrod/OK using a Genetic Algorithm“, *Procedia Computer Science* 9, 2012, pages 1647–1656.

[Lin2011] Simon C. Lin and Eric Yen, eds., *Data Driven e-Science*, Springer, 2011.

[Lindenbaum2011] Pierre Lindenbaum, Solena Le Scouarnec, Vincent Portero, and Richard Redon, „Knime4Bio: a set of custom nodes for the interpretation of next-generation sequencing data with KNIME“, *Bioinformatics* 27 (22), 2011, pages 3200–3201.

[Littauer2012] Richard Littauer, Karthik Ram, Bertram Ludäscher, William Michener, and Rebecca Koskela, „Trends in Use of Scientific Workflows: Insights from a Public Repository and Recommendations for Best Practice“, *International Journal of Digital Curation* 7 (2), 2012, pages 92–100.

[Lozano2009] Jose A. Lozano, Qingfu Zhang, and P. Larraaga, „Guest Editorial: Special Issue on Evolutionary Algorithms Based on Probabilistic Models“, *IEEE Transactions on Evolutionary Computation* 13 (6), 2009, pages 1197–1198.

[Ludascher2006] Bertram Ludäscher, Ilkay Altintas, Chad Berkley, Dan Higgins, Efrat Jaeger, Matthew Jones, Edward A Lee, Jing Tao, and Yang Zhao, „Scientific workflow management and the Kepler system", *Concurrency and Computation: Practice and Experience* 18 (10), 2006, pages 1039–1065.

[Ludascher2009a] Bertram Ludäscher, Ilkay Altintas, Shawn Bowers, Julian Cummings, Terence Critchlow, Ewa Deelman, David De Roure, Juliana Freire, Carole Goble, Matthew Jones, Scott Klasky, Timothy McPhillips, Norbert Podhorszki, Claudio Silva, Ian Taylor, and Mladen A. Vouk, „Scientific Process Automation and Workflow Management", *Scientific Data Management: Challenges, Technology, and Deployment*, Computational Science Series, Chapman & Hall, 2009, chap. 13, pages 476–508.

[Ludascher2005] Bertram Ludäscher and Carole Goble, „Guest editors' introduction to the special section on scientific workflows", *ACM SIGMOD Record* 34 (3), 2005, pages 3–4.

[Ludascher2009b] Bertram Ludäscher, Mathias Weske, Timothy McPhillips, and Shawn Bowers, „Scientific workflows: Business as usual?", *Business Process Management*, Springer, 2009, pages 31–47.

[Lukoshko2010] Andrew Lukoshko and Andrei Sokol, „UNICORE-related projects for deploying the Belarusian national grid network", *Proceedings of the 2009 international conference on Parallel processing*, Euro-Par'09, Springer-Verlag, 2010, pages 363–370.

[Maheshwari2009] Ketan Maheshwari, Carole Goble, Paolo Missier, and Johan Montagnat, „Medical Image Processing Workflow Support on the EGEE Grid with Taverna", *22nd IEEE International Symposium on Computer-Based Medical Systems*, 2009, pages 1–7.

[Marx2012] Vivien Marx, „My data are your data", *Nature Biotechnology* 30 (6), 2012, pages 509–511.

[Matthews1975] Brian W. Matthews, „Comparison of the predicted and observed secondary structure of T4 phage lysozyme", *Biochimica et Biophysica Acta* 405 (2), 1975, pages 442–451.

[McPhillips2009] Timothy McPhillips, Shawn Bowers, Daniel Zinn, and Bertram Ludäscher, „Scientific workflow design for mere mortals“, *Future Generation Computer Systems* 25 (5), 2009, pages 541–551.

[Meffert2011] Klaus Meffert, Neil Rotstan, Chris Knowles, and Ugo Sangiorgi, *JGAP - Java Genetic Algorithms and genetic programming Package*, 2011, URL: http://jgap.sf.net (Accessed: 05/13/2013).

[Mehta2012] Gaurang Mehta, Ewa Deelman, James A. Knowles, Ting Chen, Ying Wang, Jens Vöckler, Steven Buyske, and Tara Matise, „Enabling Data and Compute Intensive Workflows in Bioinformatics“, *Proceedings of the 2011 international conference on Parallel Processing - Volume 2*, Euro-Par'11, Springer, 2012, pages 23–32.

[Memon2013] Shahbaz Memon, Sonja Holl, Morris Riedel, Bernd Schuller, and Andrew Grimshaw, „Enhancing the performance of workflow execution in e-Science environments by using the standards based Parameter Sweep Model“, *Proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery*, XSEDE '13, ACM, 2013, 56:1–56:7.

[Michalewic1996] Zbigniew Michalewicz, *Genetic algorithms + data structures = evolution programs*, Springer, 1996.

[Michalski2011] Annette Michalski, Eugen Damoc, Jan-Peter Hauschild, Oliver Lange, Andreas Wieghaus, Alexander Makarov, Nagarjuna Nagaraj, Juergen Cox, Matthias Mann, and Stevan Horning, „Mass spectrometry-based proteomics using Q Exactive, a high-performance benchtop quadrupole Orbitrap mass spectrometer“, *Molecular & Cellular Proteomics* 10 (9), 2011.

[Miettinen1999] Kaisa Miettinen, *Nonlinear Multiobjective Optimization*, vol. 12, Springer, 1999.

[Minor2011] Mirjam Minor and Sebastian Görg, „Acquiring Adaptation Cases for Scientific Workflows“, *Case-Based Reasoning Research and Development*, vol. 6880, Lecture Notes in Computer Science, Springer, 2011, pages 166–180.

[Missier2010] Paolo Missier, Stian Soiland-Reyes, Stuart Owen, Wei Tan, Alexandra Nenadic, Ian Dunlop, Alan Williams, Tom Oinn, and Carole Goble, „Taverna, reloaded", *Proceedings of the 22nd international conference on Scientific and statistical database management*, Springer, 2010, pages 471–481.

[Mohammed2012] Yassene Mohammed, Ekaterina Mostovenko, Alex A. Henneman, Rob J. Marissen, André M. Deelder, and Magnus Palmblad, „Cloud Parallel Processing of Tandem Mass Spectrometry Based Proteomics Data", *Journal of Proteome Research* 11 (10), 2012, pages 5101–5108.

[Moscato1999] Pablo Moscato, „New ideas in optimization", ed. by David Corne, Marco Dorigo, Fred Glover, Dipankar Dasgupta, Pablo Moscato, Riccardo Poli, and Kenneth V. Price, McGraw-Hill Ltd., 1999, chap. Memetic algorithms: a short introduction, pages 219–234.

[Mostovenko2011] Ekaterina Mostovenko, André M. Deelder, and Magnus Palmblad, „Protein expression dynamics during Escherichia Coli glucose-lactose diauxie", *BMC Microbiology* 11 (126), 2011.

[myGrid2011] myGrid, *Increase memory allocation*, 2011, URL: http://dev.mygrid.org.uk/wiki/display/taverna/Increase+memory+allocation (Accessed: 08/01/2013).

[Namasivaya2012] Vigneshwaran Namasivayam and Jürgen Bajorath, „Multiobjective Particle Swarm Optimization: Automated Identification of Structure–Activity Relationship-Informative Compounds with Favorable Physicochemical Property Distributions", *Journal of Chemical Information and Modeling* 52 (11), 2012, pages 2848–2855.

[NCBI2009] National Center for Biotechnology Information, U.S. National Library of Medicine, *National Center for Biotechnology Information*, 2009, URL: http://www.ncbi.nlm.nih.gov/guide/sitemap/ (Accessed: 07/13/2013).

[NESC1999] National e-Science Centre, *Defining e-Science*, 1999, URL: http://www.nesc.ac.uk/nesc/define.html (Accessed: 05/13/2013).

[NSFKepler2005] National Science Foundation, *The Kepler Project*, 2005, URL: https://kepler-project.org (Accessed: 07/17/2013).

[Neri2012] Ferrante Neri and Carlos Cotta, „Memetic algorithms and memetic computing optimization: A literature review“, *Swarm and Evolutionary Computation* 2, 2012, pages 1–14.

[Nguyen2012] Trung Thanh Nguyen, Shengxiang Yang, and Juergen Branke, „Evolutionary dynamic optimization: A survey of the state of the art“, *Swarm and Evolutionary Computation* 6, 2012, pages 1–24.

[Niazi2012] Ali Niazi and Riccardo Leardi, „Genetic algorithms in chemometrics“, *Journal of Chemometrics* 26 (6), 2012, pages 345–351.

[Nicolaou2013] Christos A. Nicolaou and Nathan Brown, „Multi-objective optimization methods in drug design“, *Drug Discovery Today: Technologies* 10 (3), 2013, e427–e435.

[Nocedal2006] Jorge Nocedal and Stephen J. Wright, *Numerical optimization*, 2nd edition, Springer, 2006.

[Obst2013] Matthias Obst, *Department of Biological and Environmental Sciences, University of Gothenburg, Sweden*, Personal Communication, 2013.

[Ochoa2012] Gabriela Ochoa, Mike Preuss, Thomas Bartz-Beielstein, and Marc Schoenauer, „Editorial for the special issue on automated design and assessment of heuristic search methods“, 20 (2), 2012, pages 161–163.

[Oliveira2011] Daniel de Oliveira, Kary Ocana, Eduardo Ogasawara, Jonas Dias, Fernanda Baiao, and Marta Mattoso, „A performance evaluation of X-ray crystallography scientific workflow using SciCumulus“, *2011 IEEE International Conference on Cloud Computing*, IEEE, 2011, pages 708–715.

[Olson2008] Gary M. Olson, Ann Zimmerman, and Nathan Bos, *Scientific Collaboration on the Internet*, The MIT Press, 2008.

[openMod2013] openModeller developers, *openModeller framework*, 2013, URL: http://openmodeller.sourceforge.net/overview.html (Accessed: 06/23/2013).

[Orebaugh2006] Angela Orebaugh, Gilbert Ramirez, Josh Burke, and Larry Pesce, *Wireshark & Ethereal Network Protocol Analyzer Toolkit (Jay Beale's Open Source Security)*, Syngress Publishing, 2006.

[OReilly2009] Tim O'Reilly, *What is web 2.0*, O'Reilly, 2009.

[Paavola2005] Marjo Paavola, Sergej Olenin, and Erkki Leppäkoski, „Are invasive species most successful in habitats of low native species richness across European brackish water seas?", *Estuarine, Coastal and Shelf Science* 64 (4), 2005, pages 738–750.

[Page2012] Kevin Page, Raúl Palma, Piotr Houbowicz, Graham Klyne, Stian Soiland-Reyes, Don Cruickshank, Rafael González-Cabero, Esteban Garcia-Cuesta, David De Roure, Jun Zhao, and Jose Manuel Gómez-Pérez, „From workflows to Research Objects: an architecture for preserving the semantics of science", *Proceedings of the 2nd International Workshop on Linked Science*, 2012.

[Palit2005] Ajoy K. Palit and Dobrivoje Popovic, *Computational Intelligence in Time Series Forecasting: Theory and Engineering Applications*, Springer, 2005.

[Palmblad2013] Magnus Palmblad, *Center for Proteomics and Metabolomics, Leiden University Medical Center, Leiden, The Netherlands*, Personal Communication, 2013.

[Palmblad2002] Magnus Palmblad, Margareta Ramström, Karin E Markides, Per Håkansson, and Jonas Bergquist, „Prediction of chromatographic retention and protein identification in liquid chromatography/mass spectrometry", *Analytical chemistry* 74 (22), 2002, pages 5826–5830.

[Parejo2012] José Antonio Parejo, Antonio Ruiz-Cortés, Sebastián Lozano, and Pablo Fernandez, „Metaheuristic optimization frameworks: a survey and benchmarking", *Soft Computing* 16 (3), 2012, pages 527–561.

[Petritis2006] Konstantinos Petritis, Lars J. Kangas, Bo Yan, Matthew E. Monroe, Eric F. Strittmatter, Wei-Jun Qian, Joshua N. Adkins, Ronald J. Moore, Ying Xu, Mary S. Lipton, David G. Camp, and Richard D. Smith, „Improved peptide elution time prediction for reversed-phase liquid chromatography-MS by incorporating peptide sequence information“, *Analytical chemistry* 78 (14), 2006, pages 5026–5039.

[Pham2000] Duc Truong Pham and Dervis Karaboga, *Intelligent Optimisation Techniques: Genetic Algorithms, Tabu Search, Simulated Annealing and Neural Networks*, Springer, 2000.

[Poladian2006] Leon Poladian and Lars Sommer Jermiin, „Multi-objective evolutionary algorithms and phylogenetic inference with multiple data sets“, *Soft Computing* 10 (4), 2006, pages 359–368.

[Prodan2005] Radu Prodan and Thomas Fahringer, „Dynamic scheduling of scientific workflow applications on the grid: a case study“, *Proceedings of the 2005 ACM symposium on Applied computing*, ACM, 2005, pages 687–694.

[PRG2013] Proteome Informatics Research Group (iPRG), *ABRF iPRG-2013 Study*, 2012, URL: http://www.abrf.org/index.cfm/group.show/ProteomicsInformaticsResearchGroup.53.htm (Accessed: 06/23/2013).

[Radetzki2006] Uwe Radetzki, Ulf Leser, Svenja C. Schulze-Rauschenbach, J. Zimmermann, Jens Lüssem, Thomas Bode, and Armin B. Cremers, „Adapters, shims, and glue - service interoperability for in silico experiments“, *Bioinformatics* 22 (9), 2006, pages 1137–1143.

[Rao2009] Singiresu S. Rao, *Engineering Optimization: Theory and Practice*, Wiley, 2009.

[Reeves1993] Colin R. Reeves, „Using Genetic Algorithms with Small Populations“, *Proceedings of the Fifth International Conference on Genetic Algorithms*, Morgan Kaufmann, 1993, pages 92–99.

[ReyesSierr2006] Margarita Reyes-Sierra and Carlos A. Coello Coello, „Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art“, *International Journal of Computational Intelligence Research* 2 (3), 2006, pages 287–308.

[Rijsbergen1979] Cornelis Joost van Rijsbergen, *Information Retrieval*, 2nd edition, Butterworth-Heinemann, 1979.

[Rogers1999] Alex Rogers and Adam Prugel-Bennett, „Genetic drift in genetic algorithm selection schemes“, *IEEE Transactions on Evolutionary Computation* 3 (4), 1999, pages 298–303.

[Rosenbrock1960] Howard H. Rosenbrock, „An automatic method for finding the greatest or least value of a function“, *The Computer Journal* 3 (3), 1960, pages 175–184.

[SantanaQui2010] Luis V. Santana-Quintero, Alfredo Arias Montano, and Carlos A. Coello Coello, „A Review of Techniques for Handling Expensive Functions in Evolutionary Multi-Objective Optimization“, *Computational Intelligence in Expensive Optimization Problems*, vol. 2, Adaptation Learning and Optimization, Springer, 2010, pages 29–59.

[Shan2010] Songqing Shan and G. Gary Wang, „Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions“, *Structural and Multidisciplinary Optimization* 41 (2), 2010, pages 219–241.

[Siarry2008] Patrick Siarry and Zbigniew Michalewicz, *Advances in Metaheuristics for Hard Optimization*, Springer, 2008.

[Singh1996] Munindar P. Singh and Mladen A. Vouk, *Scientific Workflows: Scientific Computing Meets Transactional Workflows*, National Science Foundation, 1996.

[Sivanandam2007] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms*, Springer, 2007.

[Smanchat2013] Sucha Smanchat, Maria Indrawan, Sea Ling, Colin Enticott, and David Abramson, „Scheduling parameter sweep workflow in the Grid based on resource competition“, *Future Generation Computer Systems* 29 (5), 2013, pages 1164–1183.

[Smola2004] Alex J. Smola and Bernhard Schölkopf, „A tutorial on support vector regression“, *Statistics and computing* 14 (3), 2004, pages 199–222.

[Smyth2005] Gordon K. Smyth, „Limma: linear models for microarray data“, *Bioinformatics and computational biology solutions using R and Bioconductor*, Springer, 2005, pages 397–420.

[Sobieszcza1997] Jaroslaw Sobieszczanski-Sobieski and Raphael T. Haftka, „Multidisciplinary aerospace design optimization: survey of recent developments“, *Structural optimization* 14 (1), 1997, pages 1–23.

[Sonntag2012] Mirko Sonntag and Dimka Karastoyanova, „Ad hoc Iteration and Re-execution of Activities in Workflows“, *International Journal On Advances in Software* 5 (1 and 2), 2012, pages 91–109.

[Sonntag2010] Mirko Sonntag, Dimka Karastoyanova, and Frank Leymann, „The Missing Features of Workflow Systems for Scientific Computations“, *Proceedings of the 3rd Grid Workflow Workshop*, 2010, pages 209–216.

[Souza Muno2011] Mauro Enrique de Souza Muñoz, Renato De Giovanni, Marinez Ferreira de Siqueira, Tim Sutton, Peter Brewer, Ricardo Scachetti Pereira, Dora Ann Lange Canhos, and Vanderlei Perez Canhos, „openModeller: a generic approach to species' potential distribution modelling“, *GeoInformatica* 15 (1), 2011, pages 111–135.

[Starlinger2012] Johannes Starlinger, Sarah Cohen-Boulakia, and Ulf Leser, „(Re) Use in Public Scientific Workflow Repositories“, *Scientific and Statistical Database Management*, vol. 7338, Lecture Notes in Computer Science, Springer, 2012, pages 361–378.

[Stoyanovic2010] Julia Stoyanovich, Ben Taskar, and Susan Davidson, „Exploring Repositories of Scientific Workflows“, *Proceedings of the 1st International Workshop on Workflow Approaches to New Data-centric Science*, ACM, 2010, 7:1–7:10.

[Streit2010] Achim Streit, Sandra Bergmann, Rebecca Breu, Jason Daivandy, Bastian Demuth, André Giesler, Björn Hagemeier, Sonja Holl, Valentina Huber, Daniel Mallmann, Ahmen Shiraz Memon, Mohammad Shahbaz Memon, Roger Menday, Micheal Rambadt, Morris Riedel, Mathilde Romberg, Bernd Schuller, and Thomas Lippert, „UNICORE 6 – A European Grid Technology“, *High Speed and Large Scale Scientific Computing* 18, 2010, pages 157–173.

[Stropp2012] Thomas Stropp, Timothy McPhillips, Bertram Ludäscher, and Mark Bieda, „Workflows for microarray data processing in the Kepler environment“, *BMC bioinformatics* 13 (102), 2012.

[Suman2005] Balram Suman and Prabhat Kumar, „A survey of simulated annealing as a tool for single and multiobjective optimization“, *Journal of the Operational Research Society* 57 (10), 2005, pages 1143–1160.

[Sun2012] Jianyong Sun, Jonathan M. Garibaldi, and Charlie Hodgman, „Parameter Estimation Using Metaheuristics in Systems Biology: A Comprehensive Review“, *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9 (1), 2012, pages 185–202.

[Takagi2001] Hideyuki Takagi, „Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation“, *Proceedings of the IEEE* 89 (9), 2001, pages 1275–1296.

[Talbi2009] El-Ghazali Talbi, *Metaheuristics: From Design to Implementation*, Wiley, 2009.

[Tan2010] Wei Tan, Ravi Madduri, Alexandra Nenadic, Stian Soiland-Reyes, Dinanath Sulakhe, Ian Foster, and Carole Goble, „CaGrid Workflow Toolkit: A taverna based workflow tool for cancer grid“, *BMC Bioinformatics* 11 (542), 2010.

[Taylor2006a] Ian J. Taylor, Ewa Deelman, and Dennis B. Gannon, *Workflows for e-Science: scientific workflows for grids*, Springer, 2006.

[Taylor2006b] Ian Taylor, Eddie Al-Shakarchi, and Stephen David Beck, „Distributed Audio Retrieval using Triana (DART)“, *International Computer Music Conference*, 2006, pages 6–11.

[Taylor2002] Ian Taylor, Matt Shields, and Roger Philp, „GridOneD: Peer to peer visualization using Triana: A galaxy formation test case“, *Proceedings of the UK eScience "All Hands Meeting"*, 2002.

[Taylor2003] Ian Taylor, Matthew Shields, Ian Wang, and Omer Rana, „Triana Applications within Grid Computing and Peer to Peer Environments“, *Journal of Grid Computing* 1 (2), 2003, pages 199–217.

[Tenne2010] Yoel Tenne and Chi-Keong Goh, eds., *Computational Intelligence in Optimization*, vol. 7, Adaptation, Learning, and Optimization, Springer, 2010.

[TheMath1994] The MathWorks, Inc., *MATLAB - The Language of Technical Computing*, 1994, URL: http://www.mathworks.de/products/matlab/ (Accessed: 06/27/2013).

[Tiwari2007] Abhishek Tiwari and Arvind K.T. Sekhar, „Review Article: Workflow based framework for life science informatics", *Computational Biology and Chemistry* 31 (5-6), 2007, pages 305–319.

[Triguero2012] Isaac Triguero, Joaquín Derrac, Salvador Garcia, and Francisco Herrera, „A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification", *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* 42 (1), 2012, pages 86–100.

[Unger2004] Ron Unger, „The Genetic Algorithm Approach to Protein Structure Prediction", *Applications of Evolutionary Computation in Chemistry*, vol. 110, Structure and Bonding, Springer, 2004, pages 153–175.

[UoM2011] University of Manchester, UK, *wf4ever*, 2011, URL: http://www.wf4ever-project.org/ (Accessed: 06/27/2013).

[UoM2008a] School of Computer Science, University of Manchester, UK, *myGrid*, 2008, URL: http://www.mygrid.org.uk (Accessed: 07/13/2013).

[UoM2009] School of Computer Science, University of Manchester, UK, *Taverna - open source domain independent Workflow Management System*, 2009, URL: http://www.taverna.org.uk/ (Accessed:-07/16/2013).

[UoM2008b] School of Computer Science, University of Manchester, UK and the European Bioinformatics Institute, *BioCatalogue*, 2008, URL: http://www.biocatalogue.org/ (Accessed: 07/13/2013).

[UoM2007] University of Manchester, UK and University of Southampton, *myExperiment*, 2007, URL: http://www.myexperiment.org/ (Accessed: 06/27/2013).

[UoM2012] University of Manchester, UK and University of Southampton, *myExperiment alpha*, 2012, URL: http://alpha.myexperiment.org/packs (Accessed: 08/23/2013).

[USC2010] University of Southern California, *Pegasus*, 2010, URL: http://pegasus.isi.edu/ (Accessed: 05/13/2013).

[Vesterstrm2005] Jakob Vesterstrøm, „Heuristic Algorithms in Bioinformatics“, PhD Thesis, Bioinformatics Research Center, Department of Computer Science, Faculty of Science, University of Aarhus, 2005.

[Vicario2012] Saverio Vicario, Bachir Balech, Giacinto Donvito, Pasquale Notarangelo, and Graziano Pesole, „The BioVel Project: Robust phylogenetic workflows running on the GRID“, *EMBnet. journal* 18 (B), 2012, pages 77–79.

[Vouk1997] Mladen A. Vouk and Munindar P. Singh, „Quality of Service and Scientific Workflows“, *In The Quality of Numerical Software: Assessment and Enhancements*, Chapman, 1997, pages 77–89.

[Waller1963] Jean-Pierre Waller, „The NH2-terminal residues of the proteins from cell-free extracts of E. coli“, *Journal of Molecular Biology* 7 (5), 1963, pages 483–496.

[Wang2010] Jianwu Wang, Prakashan Korambath, Seonah Kim, Scott Johnson, Kejian Jin, Daniel Crawl, Ilkay Altintas, Shava Smallen, Bill Labate, and Kendall N. Houk, „Theoretical enzyme design using the Kepler scientific workflows on the Grid“, *Procedia Computer Science* 1 (1), 2010, pages 1175–1184.

[Wassink2010] Ingo Wassink, „Work flows in life science“, PhD Thesis, University of Twente, 2010.

[Wassink2009a] Ingo Wassink, Matthijs Ooms, and Paul van der Vet, „Designing workflows on the fly using e-BioFlow“, *Service-Oriented Computing*, vol. 5900, Lecture Notes in Computer Science, Springer, 2009, pages 470–484.

[Wassink2009b] Ingo Wassink, Paul E. Van Der Vet, Katy Wolstencroft, Pieter B.T. Neerincx, Marco Roos, Han Rauwerda, and Timo M. Breit, „Analysing scientific workflows: why workflows not only connect web services“, *2009 World Conference on Services-I*, IEEE, 2009, pages 314–321.

[Weise2009a] Thomas Weise, „Global Optimization - Theory and Application“, *Self-Published,* 2009, URL: http://www.it-weise.de/projects/book.pdf (Accessed: 08/26/2013).

[Weise2012] Thomas Weise, Raymond Chiong, and Ke Tang, „Evolutionary Optimization: Pitfalls and Booby Traps“, *Journal of Computer Science and Technology* 27 (5), 2012, pages 907–936.

[Weise2009b] Thomas Weise, Michael Zapf, Raymond Chiong, and Antonio J. Nebro, „Why Is Optimization Difficult?“, *Nature-Inspired Algorithms for Optimisation*, vol. 193, Studies in Computational Intelligence, Springer, 2009, pages 1–50.

[Wf4Ever Pr2012] Wf4Ever Project, *Research Object Prefix*, 2012, URL: http://purl.org/wf4ever/ro# (Accessed: 08/23/2013).

[Wfdesc2012] Wf4Ever Project, *Wfdesc Prefix*, 2012, URL: http://purl.org/wf4ever/wfdesc# (Accessed: 08/23/2013).

[Wfprov2012] Wf4Ever Project, *Wfprov Prefix*, 2012, URL: http://purl.org/wf4ever/wfprov# (Accessed: 08/23/2013).

[White2012] David R. White, „Software review: the ECJ toolkit“, *Genetic Programming and Evolvable Machines* 13 (1), 2012, pages 65–67.

[Wieczorek2009] Marek Wieczorek, Andreas Hoheisel, and Radu Prodan, „Towards a general model of the multi-criteria workflow scheduling on the grid“, *Future Generation Computer Systems* 25 (3), 2009, pages 237–256.

[Wiley2003] Edward O. Wiley, Kristina M. McNyset, A. Townsend Peterson, C. Richard Robins, and Aimee M. Stewart, „Niche modeling and geographic range predictions in the marine environment using a machine-learning algorithm“, *Oceanography* 16 (3), 2003, pages 120–127.

[Wilmarth2013] Phillip A. Wilmarth, William J. Rathje, and Larry L. David, *An unbiased comparison of peptide identification performance between SEQUEST, Mascot and X!Tandem*, Poster presented at the 61st ASMS Conference on Mass Spectrometry and Allied Topics, 2013.

[Wolpert1997] David H. Wolpert and William G. Macready, „No free lunch theorems for optimization“, *IEEE Transactions on Evolutionary Computation* 1 (1), 1997, pages 67–82.

[Wolstencro2013] Katherine Wolstencroft, Robert Haines, Donal Fellows, Alan Williams, David Withers, Stuart Owen, Stian Soiland-Reyes, Ian Dunlop, Aleksandra Nenadic, Paul Fisher, Jiten Bhagat, Khalid Belhajjame, Finn Bacall, Alex Hardisty, Abraham Nieva de la Hidalga, Maria P. Balcazar Vargas, Shoaib Sufi, and Carole Goble, „The Taverna workflow suite: designing and executing workflows of Web Services on the desktop, web or in the cloud“, *Nucleic acids research* 41 (W1), 2013, W557–W561.

[Wolstencro2009] Katy Wolstencroft, Paul Fisher, David De Roure, and Carole Goble, „Scientific Workflows“, *Research In a Connected World*, 2009.

[WMC2013] Workflow Management Coalition, *Reference Model - Workflow Management Coalition*, 2013, URL: http://www.wfmc.org/reference-model.html (Accessed: 07/10/2013).

[Wright1991] Alden H. Wright, „Genetic Algorithms for Real Parameter Optimization“, *Proceedings of the First Workshop on Foundations of Genetic Algorithms*, Morgan Kaufmann, 1991, pages 205–218.

[Xiao2010] Qing Xiao, Feiran Zhang, Benjamin A. Nacev, Jun O. Liu, and Dehua Pei, „Protein N-terminal processing: substrate specificity of Escherichia coli and human methionine aminopeptidases“, *Biochemistry* 49 (26), 2010, pages 5588–5599.

[XSEDE2013] XSEDE, *Extreme Science and Engineering Discovery Environment*, 2013, URL: https://www.xsede.org/ (Accessed: 06/05/2013).

[Yang2010] Xiaoyu Yang, Richard P. Bruin, and Martin T. Dove, „Developing an End-to-End Scientific Workflow: A Case Study Using a Comprehensive Workflow Platform in e-Science“, *Computing in Science & Engineering* 12 (3), 2010, pages 52–61.

[Yeltayeva2012] Kymbat Yeltayeva, „Usability Study of the Taverna Scientific Workflow Workbench“, Master Thesis, University of Manchester, UK, 2012.

[Yu2005] Jia Yu and Rajkumar Buyya, „A Taxonomy of Workflow Management Systems for Grid Computing“, *Journal of Grid Computing* 3 (3-4), 2005, pages 171–200.

[Yu2008] Jia Yu, Rajkumar Buyya, and Kotagiri Ramamohanarao, „Workflow Scheduling Algorithms for Grid Computing“, *Metaheuristics for scheduling in distributed computing environments*, vol. 146, Studies in Computational Intelligence, Springer, 2008, pages 173–214.

[Yu2007] Jia Yu, Michael Kirley, and Rajkumar Buyya, „Multi-Objective Planning for Workflow Execution on Grids“, *Proceedings of the 8th IEEE/ACM International conference on Grid Computing*, IEEE, 2007, pages 10–17.

[Zhao2008] Yong Zhao, Ioan Raicu, and Ian Foster, „Scientific Workflow Systems for 21st Century, New Bottle or New Wine?“, *IEEE Congress on Services-Part I*, IEEE, 2008, pages 467–471.

[Zhou2011] Aimin Zhou, Bo-Yang Qu, Hui Li, Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Qingfu Zhang, „Multiobjective evolutionary algorithms: A survey of the state of the art“, *Swarm and Evolutionary Computation* 1 (1), 2011, pages 32–49.

[Zimmermann2008] Olav Zimmermann and Ulrich H.E. Hansmann, „LOCUSTRA: Accurate Prediction of Local Protein Structure Using a Two-Layer Support Vector Machine Approach“, *Journal of Chemical Information and Modeling* 48 (9), 2008, pages 1903–1908.

[Zitzler2000] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele, „Comparison of multiobjective evolutionary algorithms: Empirical results“, *Evolutionary computation* 8 (2), 2000, pages 173–195.

[Zuluaga2013] Marcela Zuluaga, Andreas Krause, Guillaume Sergent, and Markus Püschel, „Active Learning for Multi-Objective Optimization“, *International Conference on Machine Learning*, 2013.

Schriften des Forschungszentrums Jülich
IAS Series

1. **Three-dimensional modelling of soil-plant interactions: Consistent coupling of soil and plant root systems**
   by T. Schröder (2009), VIII, 72 pages
   ISBN: 978-3-89336-576-0
   URN: urn:nbn:de:0001-00505

2. **Large-Scale Simulations of Error-Prone Quantum Computation Devices**
   by D. B. Trieu (2009), VI, 173 pages
   ISBN: 978-3-89336-601-9
   URN: urn:nbn:de:0001-00552

3. **NIC Symposium 2010**
   Proceedings, 24 – 25 February 2010 | Jülich, Germany
   edited by G. Münster, D. Wolf, M. Kremer (2010), V, 395 pages
   ISBN: 978-3-89336-606-4
   URN: urn:nbn:de:0001-2010020108

4. **Timestamp Synchronization of Concurrent Events**
   by D. Becker (2010), XVIII, 116 pages
   ISBN: 978-3-89336-625-5
   URN: urn:nbn:de:0001-2010051916

5. **UNICORE Summit 2010**
   Proceedings, 18 – 19 May 2010 | Jülich, Germany
   edited by A. Streit, M. Romberg, D. Mallmann (2010), iv, 123 pages
   ISBN: 978-3-89336-661-3
   URN: urn:nbn:de:0001-2010082304

6. **Fast Methods for Long-Range Interactions in Complex Systems**
   Lecture Notes, Summer School, 6 – 10 September 2010, Jülich, Germany
   edited by P. Gibbon, T. Lippert, G. Sutmann (2011), ii, 167 pages
   ISBN: 978-3-89336-714-6
   URN: urn:nbn:de:0001-2011051907

7. **Generalized Algebraic Kernels and Multipole Expansions for Massively Parallel Vortex Particle Methods**
   by R. Speck (2011), iv, 125 pages
   ISBN: 978-3-89336-733-7
   URN: urn:nbn:de:0001-2011083003

8. **From Computational Biophysics to Systems Biology (CBSB11)**
   Proceedings, 20 - 22 July 2011 | Jülich, Germany
   edited by P. Carloni, U. H. E. Hansmann, T. Lippert, J. H. Meinke, S. Mohanty, W. Nadler, O. Zimmermann (2011), v, 255 pages
   ISBN: 978-3-89336-748-1
   URN: urn:nbn:de:0001-2011112819

Schriften des Forschungszentrums Jülich
IAS Series

9. **UNICORE Summit 2011**
Proceedings, 7 - 8 July 2011 | Toruń, Poland
edited by M. Romberg, P. Bała, R. Müller-Pfefferkorn, D. Mallmann (2011), iv, 150 pages
ISBN: 978-3-89336-750-4
URN: urn:nbn:de:0001-2011120103

10. **Hierarchical Methods for Dynamics in Complex Molecular Systems**
Lecture Notes, IAS Winter School, 5 – 9 March 2012, Jülich, Germany
edited by J. Grotendorst, G. Sutmann, G. Gompper, D. Marx (2012), vi, 540 pages
ISBN: 978-3-89336-768-9
URN: urn:nbn:de:0001-2012020208

11. **Periodic Boundary Conditions and the Error-Controlled Fast Multipole Method**
by I. Kabadshow (2012), v, 126 pages
ISBN: 978-3-89336-770-2
URN: urn:nbn:de:0001-2012020810

12. **Capturing Parallel Performance Dynamics**
by Z. P. Szebenyi (2012), xxi, 192 pages
ISBN: 978-3-89336-798-6
URN: urn:nbn:de:0001-2012062204

13. **Validated force-based modeling of pedestrian dynamics**
by M. Chraibi (2012), xiv, 112 pages
ISBN: 978-3-89336-799-3
URN: urn:nbn:de:0001-2012062608

14. **Pedestrian fundamental diagrams: Comparative analysis of experiments in different geometries**
by J. Zhang (2012), xiii, 103 pages
ISBN: 978-3-89336-825-9
URN: urn:nbn:de:0001-2012102405

15. **UNICORE Summit 2012**
Proceedings, 30 - 31 May 2012 | Dresden, Germany
edited by V. Huber, R. Müller-Pfefferkorn, M. Romberg (2012), iv, 143 pages
ISBN: 978-3-89336-829-7
URN: urn:nbn:de:0001-2012111202

16. **Design and Applications of an Interoperability Reference Model for Production e-Science Infrastructures**
by M. Riedel (2013), x, 270 pages
ISBN: 978-3-89336-861-7
URN: urn:nbn:de:0001-2013031903

17. **Route Choice Modelling and Runtime Optimisation for Simulation of Building Evacuation**
by A. U. Kemloh Wagoum (2013), xviii, 122 pages
ISBN: 978-3-89336-865-5
URN: urn:nbn:de:0001-2013032608

18. **Dynamik von Personenströmen in Sportstadien**
by S. Burghardt (2013), xi, 115 pages
ISBN: 978-3-89336-879-2
URN: urn:nbn:de:0001-2013060504

19. **Multiscale Modelling Methods for Applications in Materials Science**
by I. Kondov, G. Sutmann (2013), 326 pages
ISBN: 978-3-89336-899-0
URN: urn:nbn:de:0001-2013090204

20. **High-resolution Simulations of Strongly Coupled Coulomb Systems with a Parallel Tree Code**
by M. Winkel (2013), xvii, 196 pages
ISBN: 978-3-89336-901-0
URN: urn:nbn:de:0001-2013091802

21. **UNICORE Summit 2013**
Proceedings, 18th June 2013 | Leipzig, Germany
edited by V. Huber, R. Müller-Pfefferkorn, M. Romberg (2013), iii, 94 pages
ISBN: 978-3-89336-910-2
URN: urn:nbn:de:0001-2013102109

22. **Three-dimensional Solute Transport Modeling in Coupled Soil and Plant Root Systems**
by N. Schröder (2013), xii, 126 pages
ISBN: 978-3-89336-923-2
URN: urn:nbn:de:0001-2013112209

23. **Characterizing Load and Communication Imbalance in Parallel Applications**
by D. Böhme (2014), xv, 111 pages
ISBN: 978-3-89336-940-9
URN: urn:nbn:de:0001-2014012708

24. **Automated Optimization Methods for Scientific Workflows in e-Science Infrastructures**
by S. Holl (2014), xvi, 182 pages
ISBN: 978-3-89336-949-2
URN: urn:nbn:de:0001-2014022000

Scientific workflows have emerged as a key technology that assists scientists with the design, management, execution, sharing and reuse of *in silico* experiments. Workflow management systems simplify the management of scientific workflows by providing graphical interfaces for their development, monitoring and analysis. Nowadays, e-Science combines such workflow management systems with large-scale data and computing resources into complex research infrastructures. For instance, e-Science allows the conveyance of best practice research in collaborations by providing workflow repositories, which facilitate the sharing and reuse of scientific workflows. However, scientists are still faced with different limitations while reusing workflows. One of the most common challenges they meet is the need to select appropriate applications and their individual execution parameters. If scientists do not want to rely on default or experience-based parameters, the best-effort option is to test different workflow set-ups using either trial and error approaches or parameter sweeps. Both methods may be inefficient or time consuming respectively, especially when tuning a large number of parameters. Therefore, scientists require an effective and efficient mechanism that automatically tests different workflow set-ups in an intelligent way and will help them to improve their scientific results.

This thesis addresses the limitation described above by defining and implementing an approach for the optimization of scientific workflows. In the course of this work, scientists' needs are investigated and requirements are formulated resulting in an appropriate optimization concept. This concept is prototypically implemented by extending a workflow management system with an optimization framework. This implementation and therewith the general approach of workflow optimization is experimentally verified by four use cases in the life science domain. Finally, a new collaboration-based approach is introduced that harnesses optimization provenance to make optimization faster and more robust in the future.

This publication was written at the Jülich Supercomputing Centre (JSC) which is an integral part of the Institute for Advanced Simulation (IAS). The IAS combines the Jülich simulation sciences and the supercomputer facility in one organizational unit. It includes those parts of the scientific institutes at Forschungszentrum Jülich which use simulation on supercomputers as their main research methodology.

IAS Series
Volume 24
ISBN 978-3-89336-949-2

JÜLICH
FORSCHUNGSZENTRUM