Future Generation Computer Systems 36 (2014) 352–362

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

# A new optimization phase for scientific workflow management systems

Sonja Holl [a,*], Olav Zimmermann [a], Magnus Palmblad [b], Yassene Mohammed [b], Martin Hofmann-Apitius [c]

[a] Jülich Supercomputing Centre (JSC), Forschungszentrum Jülich, 52425 Jülich, Germany
[b] Center for Proteomics and Metabolomics, Leiden University Medical Center, The Netherlands
[c] Fraunhofer Institute for Algorithms and Scientific Computing (SCAI) Schloss Birlinghoven, 53754 Sankt Augustin, Germany

## HIGHLIGHTS

- Proposal of a novel optimization phase to the common scientific workflow life cycle.
- Development of an automated optimization framework implementing the optimization phase.
- Plugin mechanism to support the development of arbitrary optimization methods.
- Implementation of a Genetic Algorithm-based parameter optimization plugin.
- Optimization of two use cases to demonstrate ease of use and computational efficiency.

## ARTICLE INFO

*Article history:*
Received 1 February 2013
Received in revised form
21 June 2013
Accepted 5 September 2013
Available online 23 September 2013

*Keywords:*
e-science
Workflow optimization
Taverna
Genetic algorithm
Workflow life cycle

## ABSTRACT

Scientific workflows have emerged as an important tool for combining the computational power with data analysis for all scientific domains in e-science, especially in the life sciences. They help scientists to design and execute complex *in silico* experiments. However, with rising complexity it becomes increasingly impractical to optimize scientific workflows by trial and error. To address this issue, we propose to insert a new optimization phase into the common scientific workflow life cycle. This paper describes the design and implementation of an automated optimization framework for scientific workflows to implement this phase. Our framework was integrated into Taverna, a life-science oriented workflow management system and offers a versatile programming interface (API), which enables easy integration of arbitrary optimization methods. We have used this API to develop an example plugin for parameter optimization that is based on a Genetic Algorithm. Two use cases taken from the areas of structural bioinformatics and proteomics demonstrate how our framework facilitates setup, execution, and monitoring of workflow parameter optimization in high performance computing e-science environments.



## 1. Introduction

Computational science has joined theory and experiment as a third methodology to perform science. One of the main applications of *in silico* experiments besides computer simulation is data analysis—computational processes that analyze data to gain research results. Depending on the data volume, the data types, and the task in question different combinations of computer-based methods have to be used in concert to obtain the desired analysis result. To address the challenges in set-up, execution and management of these complex *in silico* experiments, scientific workflows have become an increasingly popular choice as they allow for easy comprehension, editing, and dissemination of analysis recipes in the life science domain. In particular, they have been successfully utilized in e-science environments [1], which provide researchers access to large scale computational and data resources and enable seamless and secure collaborations.

Originating from business workflows [2], important aspects of scientific workflows are workflow enactment, modeling and execution as well as sharing. As they precisely define the execution and data flow, scientific workflows enable the management of even heterogeneous and complex scientific computing experiments. A successful approach that has emerged, organizes the development of such *in silico* experiments in a scientific workflow life cycle [3]. This concept describes the cyclic process by which a workflow passes through several phases including design, planning, sharing, execution, analysis, and learning, until the goal of the *in silico* experiment has been achieved. However, the increasing complexity

* Corresponding author. Tel.: +49 2461 612760.
*E-mail address:* s.holl@fz-juelich.de (S. Holl).

0167-739X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.future.2013.09.005

of scientific workflows, in particular in the life sciences, poses new challenges such as keeping track of applications, interrelations, and dependencies. For complex workflows choosing parameters *ad hoc* or using the default parameters of the applications will often be suboptimal and may lead to poor results or even failure of the entire *in silico* experiment.

A scientific workflow typically consists of a set of components, each representing a scientific application, which are logically connected to define a specific data flow. The parameters of these components can have complex interrelationships and some components will have a stronger impact on the final result than others. To allow for iterative improvement of these factors, the established scientific workflow life cycle provides a learning phase after the execution [3–5]. In this phase, researchers analyze the results and refine the workflow, e.g. by changing parameters before continuing the life cycle to obtain potentially better results. While for smaller workflows it may be feasible to find near optimal workflow settings using *trial and error*, such manual optimization becomes impractical for complex workflows, due to the large number of possible choices for parameters, components and workflow topology. Furthermore, we argue that this approach is not very resource-efficient as it requires that the whole workflow is executed while typically only parts of a workflow require refinement. Instead, we propose to insert a new phase into the scientific workflow life cycle which performs the scientific workflow optimization more efficiently and in an automated manner.

The typical scientific workflow life cycle is supported by various scientific workflow management systems [3]. They ease the design, creation, automated execution and result analysis of scientific workflows. Some optimization approaches have already been implemented for these workflow management systems in the life science domain. For example, Kumar et al. [6] designed an integrated framework for parameter optimizations, but with particular focus on runtime performance optimizations. Abramson et al. [7] developed a tool that aims at parameter optimization for models and inverse problems. In another approach [8], the authors defined a fixed optimization workflow to optimize the application parameters. In contrast, our work describes the extension of a workflow management system (WMS) by a general optimization framework that can flexibly deal with many types of automated scientific workflow optimizations. By developing a general framework, we want to ease the implementation and integration of optimization methods into WMS for developers. As optimizations may become very compute intensive, we focus in our work on parallel and distributed approaches for workflow optimization.

A shorter version of our approach has been presented at the IEEE International Conference on E-Science [9]. In the present contribution we show the general aspects of our optimization framework for the life science domain, outline potential optimization levels, present an example implementation of a plugin for optimization at the parameter level and added a real world use case representing a typical analysis task for a proteomics lab.

In Section 2, we give a detailed description of the common scientific workflow life cycle, details of our new automated optimization phase and its generic approach, including potential scientific workflow optimization levels. Section 3 describes our main contribution, an optimization framework embedded into an established scientific workflow management system. Additionally, this section provides details of our parallel and distributed approach. Section 4 describes the implemented example plugin for optimization at the parameter level, explaining both the developer perspective and the user perspective of our framework. Finally, two life science use cases taken from structural bioinformatics and proteomics are exposed in Section 5. Section 6 compares our work to the related approaches and Section 7 finally concludes the paper.

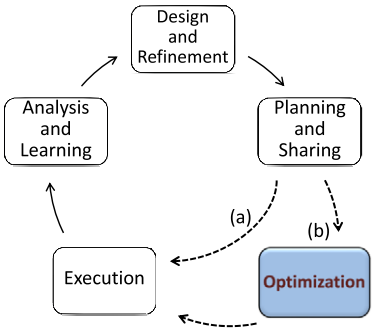


**Fig. 1.** (a) The well described scientific workflow life cycle. (b) The scientific workflow life cycle extended by the new optimization phase.

## 2. The scientific workflow life cycle

### 2.1. State-of-the-art

State-of-the-art development for an *in silico* experiment in the life sciences is a cyclic process with several steps. In the literature the term workflow life cycle [2] has been coined for this development process and several successful applications to scientific workflows have been described [3–5,10,11]. The definitions of the life cycle may vary in their particular specifications but generally lead to the same architecture shown in Fig. 1(a). The individual phases are described in the following.

*Design and refinement*

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. Although in some definitions, creation of an executable workflow from an abstract template belongs to the 'design and refinement' phase, which we will describe it as part of the 'planning and sharing' phase.

*Planning and sharing*

The description of the 'planning and sharing' phase in the literature is very heterogeneous. 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 as well as execution resources are selected. A thorough planning is particularly important for large scale and compute intensive workflows (applications) as in these cases mappings to high performance computing (HPC), Grid or Cloud resources have to be precisely defined. 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.

*Workflow execution*

Workflow execution is typically managed by a workflow engine. This engine maps the executable to an appropriate execution environment by retrieving information about available software, computing resources and data resources. 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*

To successfully evolve the scientific experiment, the scientific workflow life cycle contains as the *last* phase an analysis and learning step. Some authors also include into this phase the publishing