On Specifying and Sharing Scientific Workflow Optimization Results Using Research Objects

Sonja Holl
Jülich Supercomputing Centre
Forschungszentrum Jülich
52425 Jülich, Germany
s.holl@fz-juelich.de

Daniel Garijo
Ontology Engineering Group
Facultad de Informática
Universidad Politécnica de Madrid
dgarijo@fi.upm.es

Khalid Belhajjame
School of Computer Science
University of Manchester, UK
Khalid.Belhajjame@cs.man.ac.uk

Olav Zimmermann
Jülich Supercomputing Centre
Forschungszentrum Jülich
52425 Jülich, Germany
olav.zimmermann@fz-juelich.de

Renato De Giovanni
Reference Center on Environmental Information
Campinas SP, Brazil
renato@cria.org.br

Matthias Obst
Department of Biological and Environmental Sciences
University of Gothenburg, Sweden
matthias.obst@bioenv.gu.se

ABSTRACT

Reusing and repurposing scientific workflows for novel scientific experiments is nowadays facilitated by workflow repositories. Such repositories allow scientists to find existing workflows and re-execute them. However, workflow input parameters often need to be adjusted to the research problem at hand. Adapting these parameters may become a daunting task due to the infinite combinations of their values in a wide range of applications. Thus, a scientist may preferably use an automated optimization mechanism to adjust the workflow set-up and improve the result. Currently, automated optimizations must be started from scratch as optimization meta-data are not stored together with workflow provenance data. This important meta-data is lost and can neither be reused nor assessed by other researchers. In this paper we present a novel approach to capture optimization meta-data by extending the Research Object model and reusing the W3C standards. We validate our proposal through a real-world use case taken from the biodivertsity domain, and discuss the exploitation of our solution in the context of existing e-Science infrastructures.

Keywords

Scientific Workflows, Optimization, Research Object, Ontology, Taverna

1. INTRODUCTION

In recent years scientific workflows have emerged as an alternative to script programming for performing in-silico experiments. Scientific workflows describe the set of tasks needed to carry out a computational experiment [5]. Similar to the 'mashup' concept in web programming, many scientific ideas can be phrased as different combinations of existing algorithmic building blocks (also known as components). Scientific workflows have been particularly attractive to scientists aiming to expose, share and reuse their work [14], as they help specifying the methods used for each step of the experiment. Consequentially, e-Science environments have started to provide public repositories for collection and sharing of scientific workflows [20] thereby providing a source for large amounts of material for assembling novel methods.

However, even if a researcher has successfully assembled a workflow, it is often necessary to find suitable parameters of the workflow components for its execution. This is not a trivial task, since the quality of the final result of an experiment depends on the choices for these input parameters. Therefore, when sharing and reusing scientific workflow results, the final choice of the input parameters should be justified.

In order to find a suitable parameter set, scientists frequently use trial and error or parameter sweeps. Lately, they may also use optimization techniques provided for scientific workflows [9]. This optimization process has to be performed for each workflow from scratch and is neither captured nor shared among scientists.

However, the optimization process could be much more simplified and improved if public workflow repositories stored not only provenance data of the workflow execution but also the necessary meta-information of the optimization process itself. This optimization provenance could be used to refine further optimization runs. For instance, it could help to identify those parts in the parameter value search space that do not need to be searched again because the fitness values of prior runs were poor. Another way would be to use a set of parameter and fitness values for the first breeding process. Provenance reuse may allow for a faster convergence of the optimization algorithm and is likely to improve the workflow overall results. Researchers could then reuse the suitable results of others to automatically improve their own scientific result and reduce the time required for analysis. Additionally, the computing time required for the optimization process is wasted if the optimization results were not stored and reused.

WORKS13, November 17, 2013, Denver, CO, USA

ACM 978-1-4503-2502-8/13/11 ...$15.00
http://dx.doi.org/10.1145/2534248.2534251.

In this paper we propose a novel approach to capture workflow optimization meta-data, which is validated against a real world use case. In the following sub-section we motivate the required concept.

1.1 Contributions

Rather than developing our solution from scratch, we have built our approach based on the Research Object model [1], already aligned with the W3C standards.

The Research Object model was developed to promote the sharing and preservation of research artifacts, in particular those involving scientific workflows. It supports the description of the scientific processes in a machine processable format, together with the datasets involved, the results obtained, and their provenance information.

The resources that compose a Research Object, as well as the Research Object itself are accompanied by annotations, which promote the discoverability, and therefore the reusability of the workflows, as well as enabling third parties to assess the validity and reproducibility of the results. The model is implemented in the form of a family of ontologies, where each ontology captures a specific facet of the scientific experiment.

Our model captures the specific aspects of workflow optimizations. Specifically, we make the following contributions:

- **A Novel Optimization Ontology (RO-Opt)** that extends the Research Object ontologies for capturing and sharing workflow optimizations.
- **A Real-World Case Study for Capturing Workflow Optimization**. As a proof of concept, we show how the proposed optimization ontology was used to capture the optimization results of a real world scientific workflow from the biodiversity domain.

The paper is organized as follows. A background for the motivation of workflow optimization is presented in Section 2. We introduce the concepts of scientific workflows and workflow optimization in Section 3. We present the ontology that we designed for capturing workflow optimizations in Section 4. In Section 5, we present the case study showing how the ontology was used to capture the optimization results of a real world workflow for ecology niche modeling. We discuss the use of the ontology proposed in the context of an e-Science infrastructure such as myExperiment [7] in Section 6. Finally, we conclude the paper underlining our contributions and discussing future work in Section 7.

2. SCIENTIFIC WORKFLOW OPTIMIZATION: BACKGROUND

This section provides background information and related work to the reader in order to underpin our motivation to support scientific workflow optimization.

The "Golden Trail" project [17] aims to build a provenance infrastructure for workflow traces. Special focus is put on those traces, which represent *best practice* results for scientists and how these results evolved. This is pursued by merging historically related provenance traces. By querying the workflow repository, a scientist can receive a *best practice* workflow and analyze the traces to see how this set-up evolved. This may give an idea of the parameter distribution, but requires a certain number of available traces to form a hypothesis. Although the found workflow is *best practice*, it depends on the applied input data and may not be sufficiently tested and explored. Even if a parameter study was performed, the resulting trace may become difficult to follow and the researcher may not be able to extract important information to improve his own workflow set-up.

In order to generally assist workflow users in searching suitable values for scientific workflows, several researchers have explored the usage of optimization techniques. As a result, some tools have been made available to enhance the performance of the so called *parameter sweep workflows* [12, 2, 19]. These parameter sweeps are used to systematically test parameters and find a best parameter set. The parameter settings obtained using parameter sweeps improve the workflow output and thus the scientific result. However, parameter sweeps may be computational intensive or sample the search space in an ineffective manner. Because of this, authors of the present paper have recently developed a general framework for automated parameter optimization of scientific workflows [9]. This framework is extendable and uses the knowledge provided by the researcher to narrow the parameter space via constraints or parameter dependencies. Based on this, the search space is limited and can be intelligently searched via heuristic optimization methods.

However, the individual workflow executions are currently stored separate from each other and all meta-data, such as the used optimization algorithm or parameter ranges, are lost. Currently, the ordinary captured workflow provenance of optimized workflow will not give any improvement with respect to non-optimized workflow provenance. In this paper we focus on a method to capture the optimization meta-data.

3. PRELIMINARIES

Workflow optimization is integrated in the common scientific workflow life cycle. In order to provide an appropriate context of workflow optimization, this section describes the extended scientific workflow life cycle and the optimization process in detail.

3.1 Life Cycle of Scientific Workflows

The development of a scientific workflow can be organized in a cyclic process with several steps including optimization [9] as illustrated in Figure 1. The individual phases including the novel optimization phase, which are further explained in [9], are summarized below.

Design and Refinement.

The initial cycle usually starts with the design of a new workflow or the refinement of an existing workflow taken from a workflow repository. During this phase the components of the workflow are designed, representing the single steps of an experiment. Afterwards, the composition of these components is established, including the precise definition of the dependencies between data and components.

Sharing and Planning.

In general, this phase is used to share the designed workflow with the community in an e-Science infrastructure. The aim is that other researchers can access the workflows to run or extend them. Planning refers to turning the abstract workflow created during design phase into a concrete executable workflow. This is achieved by mapping abstract parts to

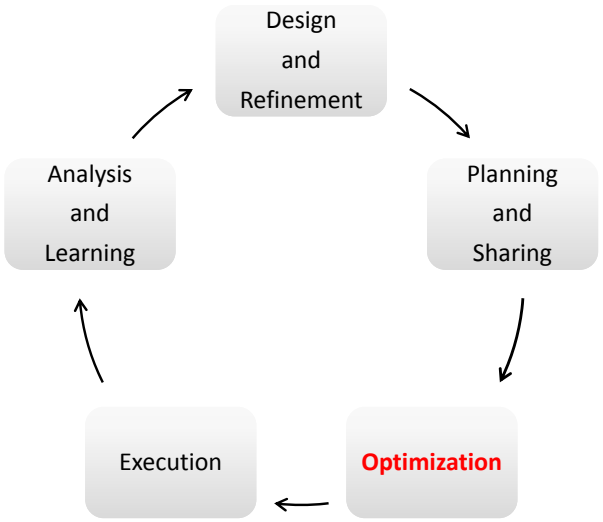

Figure 1: The extended scientific workflow life cycle showing the common cycle including the novel integrated optimization phase.

concrete components of the workflow. Parameters and data sources are defined as inputs and execution resources are selected.

Optimization.

During this phase, the scientific workflow is optimized in an automated way with the aim of identifying the values which yield the best results. Typically, only a user defined sub-workflow is optimized in order to avoid executing the entire workflow and including components that do not affect the result. The definition of a better workflow result is application-dependent and is defined by the user.

Workflow Execution.

The workflow execution is typically managed by a workflow engine. The engine maps the execution to an appropriate execution environment by retrieving information about suitable 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 execution are then sent back to the engine, and passed on to the user.

Analysis and Learning.

In order to successfully elaborate the scientific experiment, the scientific workflow life cycle contains a last phase for analysis and learning. Some definitions also include into this phase the publishing of the workflow and the results [15, 6]. The phase includes the examination as well as a comparison of the obtained results with those of other experiments. Commonly scientists restart the life cycle after the analysis step if results do not yet match the goals or expectations. During subsequent cycles, the workflow is refined and improved using *trial and error* approaches.

3.2 Optimization of Scientific workflows

In our prior work [9] we introduced a new phase to the scientific workflow life cycle, the optimization phase, described briefly in Section 3.1. This phase performs the automated optimization of a scientific workflow. This enables the researcher to concentrate on the problem at hand rather than dealing with details of the workflow execution or manual improvement of the workflow set-up.

The optimization phase was prototypically implemented as a plugin for Taverna [18], a commonly used scientific workflow management system. As workflow optimization may target different levels of the workflow description and the optimization process can be performed by different optimization algorithms, the implemented solution was designed as an extensible optimization framework. Extensibility enables developers to implement (novel) optimization algorithms or levels as plugins and plug these into the framework. As the framework provides general methods required for workflow optimization, such as a graphical user interface (GUI), sub-workflow creation, security methods, parallel workflow execution and monitoring the developer can concentrate on the optimization algorithm itself. Accordingly, developers do not have to design a bottom-up solution each time. The proposed optimization framework can be extended by all kinds of optimization plugins.

The optimization phase was designed to perform the optimization on a sub-workflow. This sub-workflow has to be selected by the user. The selection is made via the provided GUI of the optimization framework within the Taverna Workbench. The user can select the components to be optimized and thus comprise the sub-workflow. The sub-workflow data structure is created by the optimization framework in the background hidden from the user.

In our prior work [9] we developed an exemplary optimization plugin for parameter optimization. This extension was plugged into the described optimization framework. It implements a Genetic Algorithm to sample the parameter search space of the sub-workflow. During the optimization process several different sub-workflow set-ups are tested automatically. The parameter values are intelligently chosen by the Genetic Algorithm utilizing a combination of different natural operators [10]. The number of optimization cycles depends on the researchers' decision. Among others, the execution time or the reached fitness value can serve as a termination criterion for the algorithm. After a user specified termination criterion the optimization process finishes and returns the value combination that yielded the best result.

Similar to general optimization methods, workflow optimization requires a measure to rate the individual workflow executions (the result). Therefore, the scientist has to define one or several workflow output ports that should be subject

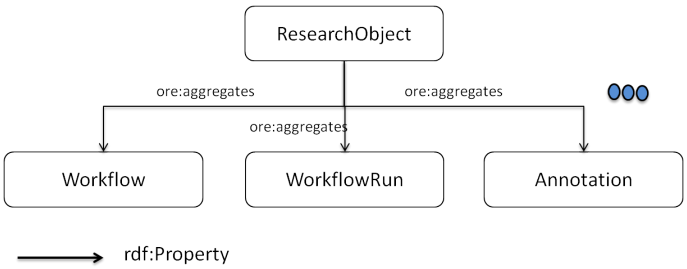

Figure 2: Research Object in a nutshell. The blue dots represent resources of other kinds of objects.

to the fitness measure. Example measures are the area under the curve or the squared correlation coefficient that is defined as one output of a component.

In order to save execution time and obtain the best result in a reasonable amount of time, the search space should be narrowed. The framework provides a specification window within its GUI, which can be implemented by the respective plugin to capture the available knowledge of the user. The exemplary parameter optimization plugin includes among other things minimum and maximum values for input parameters and dependency descriptions. The Genetic Algorithm uses only values within the user specified ranges to sample the specific parameters for workflow execution. For a more detailed description of the optimization framework, such as an architecture explanation and a screenshot, please refer to [9].

4. WORKFLOW OPTIMIZATION ONTOLOGY

In order to capture workflow optimizations, their context and their provenance (algorithms used, sub-workflows on top of which the optimization has been specified, parameters selected, etc.) we have created the generic Workflow Optimization Ontology (RO-Opt)[1]. RO-Opt is built on top of the Research Object model, reusing and extending its main concepts when necessary. In this section we first explain the basic concepts of the Research Object model in Section 4.1 and second describe the details and design decisions of the generic RO-Opt in Section 4.2.

4.1 The Research Object Model: An Overview

Research Objects [1] 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.

Figure 2 illustrates a coarse-grained view of the Research Object model. Here we focus on Workflow-Centric Research Objects, i.e., Research Objects that contain at least a workflow. A Research Object aggregates 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* that identifies a given execution of a workflow. Workflow runs in Research Objects are accompanied with provenance information specifying, amongst other things, the inputs used to feed the execution of the workflow, the intermediary steps of its internal steps as well as the results obtained at the end of the workflow execution.
- 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, e.g., 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[2], which is divided into a Research Object Core

[1] `http://purl.org/net/RO-optimization#`

[2] `http://wf4ever.github.io/ro-primer/`

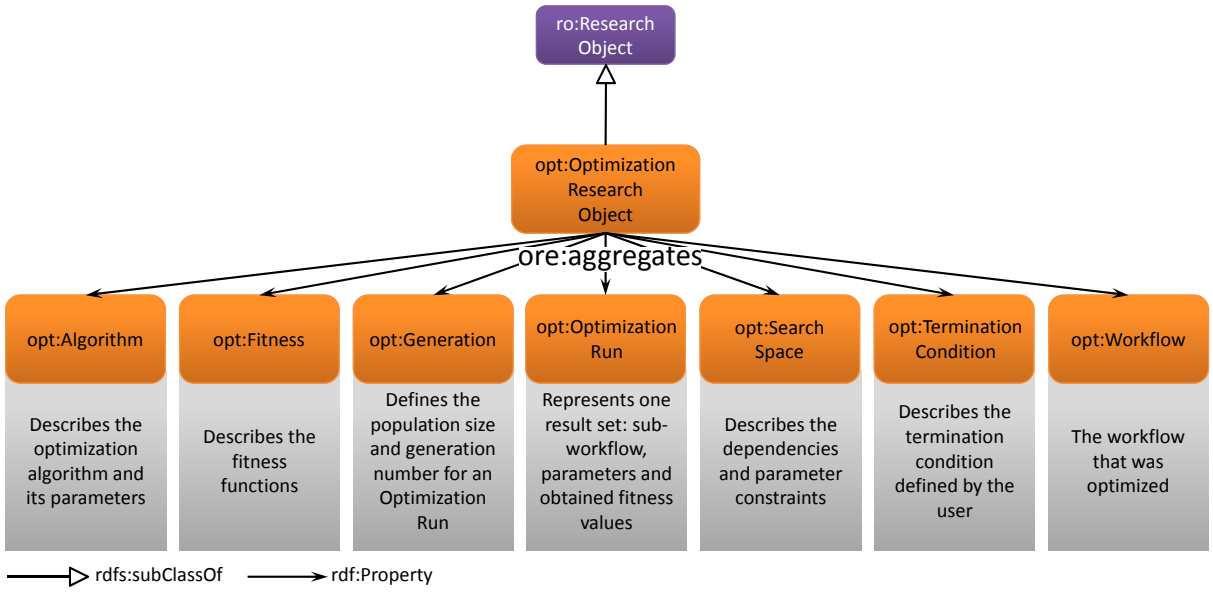

Figure 3: The main classes of the Optimization Research Object Ontology developed to store optimization provenance.

Ontology [3] and extension modules that cater to different domain specific requirements (Research Object evolution, scientific workflows, etc.). RO-Opt extends the Research Object Core Ontology and the ontologies used to specify Workflow-Centric Research Objects: the *wfdesc* ontology[4] (used to specify workflow templates) and the *wfprov* ontology[5] (used to capture the provenance traces of the workflow executions).

4.2 The Research Object Optimization Ontology (RO-Opt)

An Optimization Research Object (`opt:OptimizationResearchObject`)[6] is an aggregation of all the resources required for performing the generic workflow optimization process presented in Section 3.2. Since Research 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 3 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 at a given Generation (i.e. a given iteration) of the optimization algorithm (`opt:Generation`), the results of the **optimization run** (`opt:OptimizationRun`) as a combination of input parameters and fitness values, 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 are described in detail.

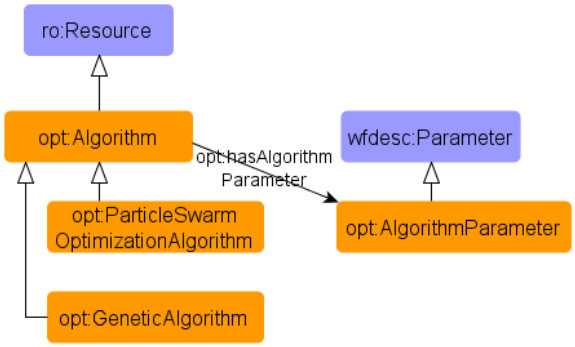

Figure 4: Classes and properties of the Algorithm section of RO-Opt. Blue concepts show the terms reused from the Research Object Ontologies.

4.2.1 *Algorithm*

As the fitness landscape of optimization problems can be rugged and contain no gradient information [21], meta-heuristic search methods [3] are convenient search algorithms (`opt:Algorithm`) to deal with the optimization of scientific workflows. Figure 4 shows the type of search algorithms we focus on this paper: Genetic Algorithms (`opt:GeneticAlgorithm`) and Particle Swarm Optimization (`opt:ParticleSwarmOptimizationAlgorithm`). As workflow optimization is an ongoing research topic, other algorithms can be plugged into the optimization framework. In such case, the ontology requires to be extended for the algorithm class respectively. Offering such a generic mechanism allows the fast extension of arbitrary optimization algorithms.

The used optimization algorithms should be stored for identification and comparison. Not only the type of algorithm but also the specific parameters of the algorithm may be reused in later optimization runs.

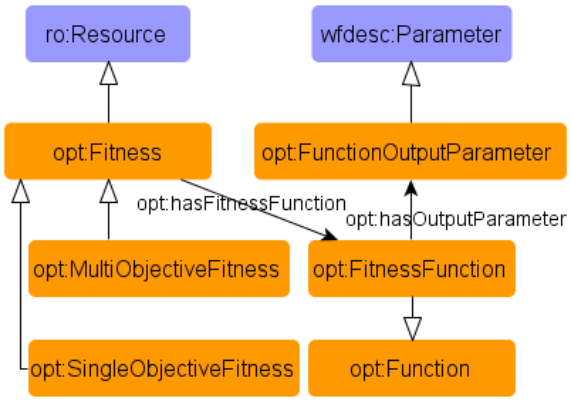

Figure 5: Classes and properties of the Fitness section of RO-Opt.

4.2.2 *Fitness*

In order to automate the optimization process, workflow results are evaluated by the optimization algorithm. The workflow results are represented by output parameters (`opt:FunctionOutputParameter`), which represents a specific fitness measure (`opt:Fitness`). Figure 5 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`). Multi-objective optimization tries to solve problems that have two or more, often conflicting, objectives [16]. Single-objective optimization in turn tries to solve only a single objective. The ontology can capture both types of optimization objectives. 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.

4.2.3 *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.

[3]Prefix ro: `http://purl.org/wf4ever/ro#`
[4]Prefix wfdesc: `http://purl.org/wf4ever/wfdesc#`
[5]Prefix wfrpov: `http://purl.org/wf4ever/wfprov#`
[6]Prefix opt: `http://purl.org/net/RO-optimization#`

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 for each generation in particular.

4.2.4 Optimization Run

Results obtained during the optimization process may be of interest and required to learn from them in later optimization runs (`opt:OptimizationRun`). Thus the original executed workflow instances and their results should be captured as well. Most scientific workflow management systems allow to export a trace of the workflow execution and results. However, we recommend against storing the entire provenance traces of several optimization workflow runs if one optimization trace has already been saved. This is due to storage limitations, since many relevant data objects (e.g. intermediate results) may require a lot of space. For example, capturing the provenance of one execution of the use case presented in Section 5 produced files of a total of 7.7MB in disk. These data objects are often negligible when reproducing an optimization and especially when learning from optimization runs. The crucial values to associate to the optimization run (`opt:OptimizationRun`) are the fitness values (`opt:hasFitnessValue`) and a flag (`opt:Flag`), which depicts the origin of the fitness value. In our example, by capturing just input and output values produced a data file of 9.6kB. As shown in Figure 6, the flag indicates that the fitness value may have been calculated (`opt:Original`), approximated (`opt:Approx`) or just taken from a prior similar or identical optimization run (`opt:LinkToOriginal`). The flag allows users to relate where the specific fitness value originates from. In particular, when optimizing identical workflows it would be useful to reuse fitness values from prior optimization runs (and use the `opt:LinkToOriginal` flag).

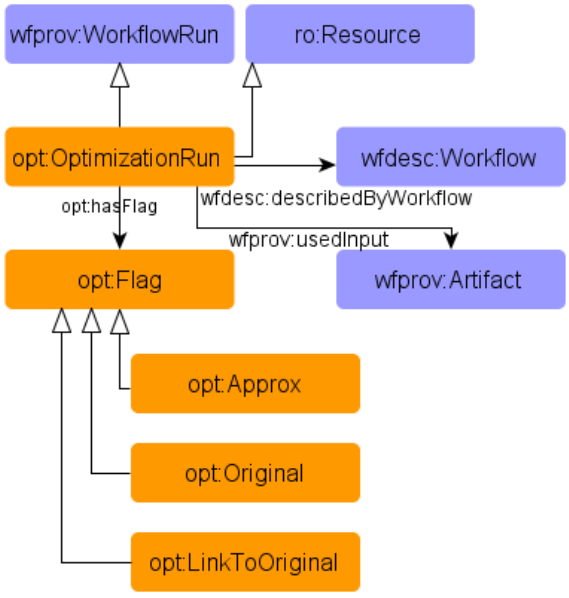

Figure 6: Classes and properties of the Optimization Run section of RO-Opt.

4.2.5 Search Space

Each workflow variation represents one specific set-up and can be sampled within 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 (i.e. processors) are valid, the number of tested workflow set-ups can be reduced by sampling the search space with 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 Figure 7, the search space comprises component and parameter dependencies respectively (`opt:ProcessorDependecy` and `opt:ParameterDependency`) as well as parameter constraints (linked with the data property `opt:hasParameterConstraint`).

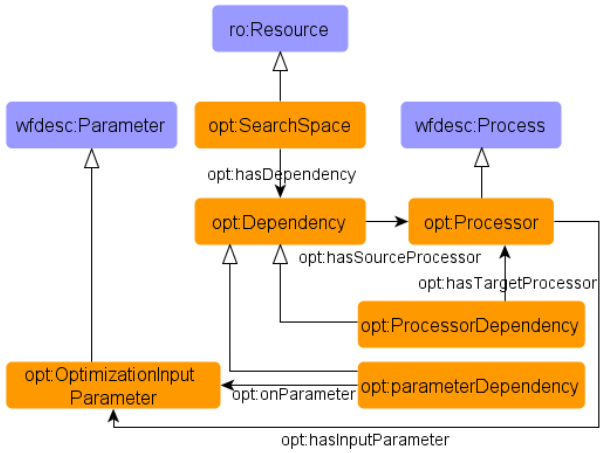

Figure 7: Classes and properties of the Search Space section of RO-Opt. The specific types of input parameters (numeric, string, etc.) and data properties have been ommitted for simplicity.

Constraints limit possible values for parameters. Whereas numerical parameters (double or integer) may be limited by a minimum and maximum value, other parameter types may also be limited. As an example, a parameter can be defined by a regular expression or by a fixed list of valid values. Parameters and components might also have specific dependencies. For example, a parameter A can be dependent on another parameter B by e.g. a numeric sum dependency such as $A+B=1$. In a similar manner, workflow components can be dependent on each other. Consider for example the following component dependency: a task can be performed by component D or A and another task by component B or C. Additionally, component A can only be executed together with component C but not with component B. These dependencies can be asserted with the `opt:onParameter` and `opt:hasTargetProcessor` properties.

4.2.6 Termination Condition

The optimization process will stop at a certain time preferably when the algorithm has converged. However, as this may be a very time consuming task, the user often wants to add additional termination criteria. The termination condition (`opt:TerminationCondition`) should be stored due to reasons regarding the plausibility of the optimization

run. The termination condition precisely stores under which condition the optimization is to be terminated. This information allows the user to verify why the optimization process ended and whether it affected the overall result or not. It 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, inferences can be made about the value of the best recorded result.

4.2.7 *Workflow*

The original workflow (`wfdesc:Workflow`) should be stored within the Optimization Research Object to increase the optimization process discoverability (with the `opt:hasWorkflow` property). If a researcher wants to optimize a similar or identical workflow, the original workflow can be used to search the repository and find similar structured workflows. 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 the input data from the non-optimized input ports.

4.2.8 *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 [16] are stored.

The presented Research Object optimization ontology can be used to capture the optimization process of scientific workflows. The ontology was designed to be generic and usable for many different scenarios. In order to support further optimization algorithms, a new sub-class of `opt:Algorithm` has to be extended. All other entities and relations were defined to be generic so they can be reused by any optimization algorithm or optimization level.

In the next section we show an example use case that was optimized and where the optimization provenance was captured manually to show the usage of our developed ontology.

5. THE ECOLOGICAL NICHE MODELING WORKFLOW

The BioVeL[7] project is building a virtual e-laboratory which includes the development of scientific workflows within the ecological domain. Among others, an Ecological Niche Modeling (ENM) workflow was developed in Taverna [18], which performs the analysis of species distributions and predicts changes in biodiversity patterns [22, 8, 13]. 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), can be used to predict the persistence of a species [11]. Thus,

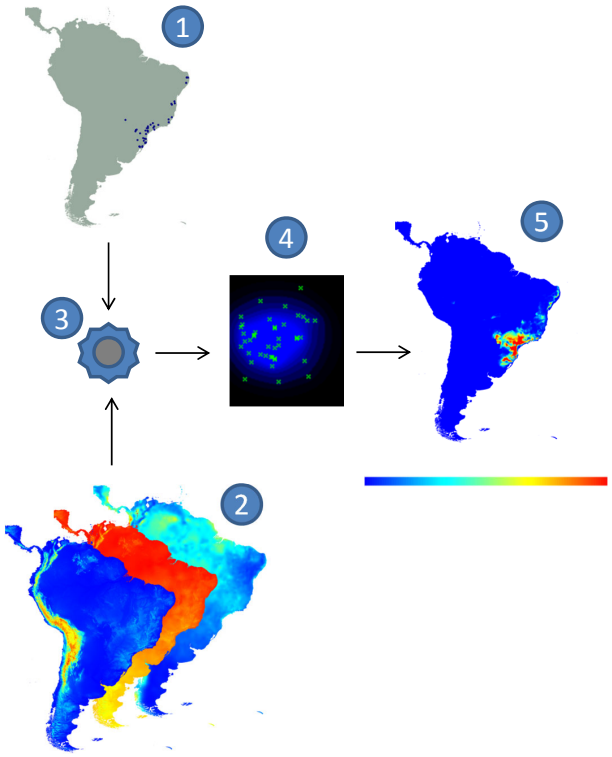

Figure 8: General principle of Ecological Niche modeling. (1) Species occurrence points. (2) Set of spatially explicit environmental variables that influence the distribution of the species. (3) Ecological niche modeling algorithm. (4) Generated model in the environmental space. (5) Model projected back into the geographical space, showing habitat suitability for the species from blue (unsuitable) to red (suitable)[9].

potential distribution models can be generated with relatively few variables characterizing the abiotic environment of the species in the form of geo-referenced raster layers (Figure 8).

In order to obtain valid models for species niches, in many cases it is important to specify the appropriate set of parameter values for a given input data set (occurrences, mask, and layers). The best parameter values can vary between different input data sets and hence they are difficult to know beforehand. We used the aforementioned optimization framework and parameter optimization plugin [9] in the workflow management system Taverna [18]. As the BioVeL ENM workflow was originally built using Taverna, further efforts for reuse were not required.

The workflow uses web services to remotely execute a specific modeling algorithm, provided by openModeller[10] [4]. An abstract description of the workflow is shown in Figure 9. Within the first step, 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

[7] `http://www.biovel.eu`

[9] Figure source: `http://openmodeller.sourceforge.net/overview.html`

[10] `http://openmodeller.sf.net/`

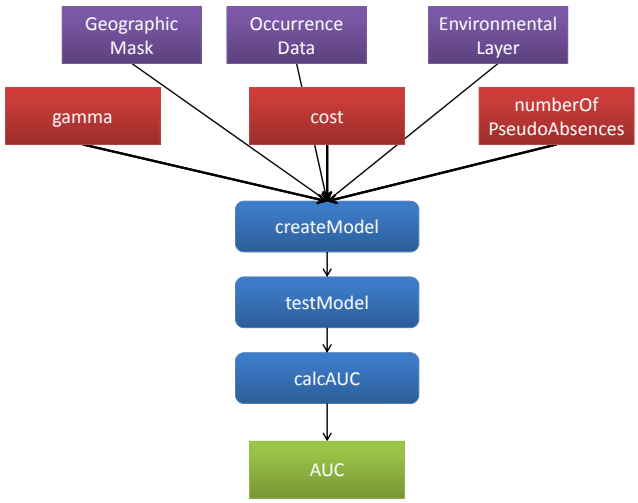

Figure 9: The abstract ENM workflow uses occurrence and environmental data to model ecological niches based on a variety of algorithms, including Support Vector Machines and others. The purple squares represent fixed parameters, the red the optimized parameters, blue the components and green the output used as fitness function.

(ROC) curve and the area under the curve (AUC). A 10-fold cross-validation was used to test model prediction, each time measuring the AUC and in the end using the average AUC as the single-objective fitness measure to optimize the ENM workflow.

As an example use case, we took the ENM workflow, using a support vector machine (SVM) algorithm to calculate the model[11]. The type of SVM that was chosen has among other configurable parameters the following: *gamma*, *cost* and the number of pseudo absences, since only presence points were used as input and the type of SVM that was chosen requires two classes for training (when no absences are specified, the algorithm internally generates an indicated number of pseudo absences). These parameters were identified by the workflow developers as those that can affect model results for the selected type of SVM and kernel function. To limit the search space of the optimization problem, various ranges were set to these parameters to limit the search space. More precisely, *gamma* was determined to range from 0 to 10, *cost* from 0 to 256 and *numberOfPseudoAbsences* from 200 to 600. Gamma and *cost* were defined as a double value while *numberOfPseudoAbsences* was defined as an integer values. To describe the *cost* parameter, we further restricted cost to be a power of 2, between 2^0 and 2^8 (which corresponds to 0 to 256). These range descriptions can be entered through the graphical user interface of the optimization framework in Taverna. A dataset from the algae: *Prorocentrum minimum* was used [12] was used as input dataset.

The increase in frequency and intensity of *Prorocentrum minimum* 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 consequences for the tourism sector. Although toxic algal blooms represent a serious public health and economic problem, no comprehensive forecasting systems for *Prorocentrum minimum* is in place for research and management.

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 and mean photosynthetically available radiation).

The optimization process was performed based on this input data and the previously defined constraints. Initially, the population size was set to 16 and the total runtime was limited to 24h due to third party restrictions. The best fitness (AUC) obtained was 0.9207 with $gamma = 2.36$ $cost$=2^3 and $numberofPseudoAbsences = 363$.

This optimization run has been captured with the RO-Opt ontology. In doing so, the described ontology was applied to manually model the performed optimization process as an OptimizationResearchObject and thus create optimization provenance. This model stores the three parameters modified during the optimization run, their constraints and the termination criterion: 24 hours maximum execution time. All workflow instances as well as the best result have been recorded. Figure 10 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 complete Optimization Research Object is available online[13].

Since this provenance meta-data is available, a second optimization run could e.g. restore the tested parameters and obtained fitness values and take them as granted during the optimization. This run could then reuse already obtained gradient information and sample the next parameter values in a promising area.

6. TOWARDS SHARING AND EXPLOITING OPTIMIZATION RESEARCH OBJECTS

In the previous section we showed how to manually create and store optimization provenance data. 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 set, as the optimization algorithm can include the results during further optimization runs of the same workflow and data. The algorithm may thereby identify gradient information and converge to a (local) optimum more quickly. If different but similar data is used, a researcher can explore prior optimization runs to gain insights into relevant parameters and their ranges. Statistics about the used parameter space may help identify relevant parameters or parameter ranges.

Workflow optimization and the presented ontology would benefit from a community-wide adoption and reveal 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. In this respect, it is worth mentioning that the development

[11] `http://www.myexperiment.org/workflows/3680.html`
[12] Dataset available from `http://www.gbif.org`
[13] `http://purl.org/net/svm-opt-research-object`

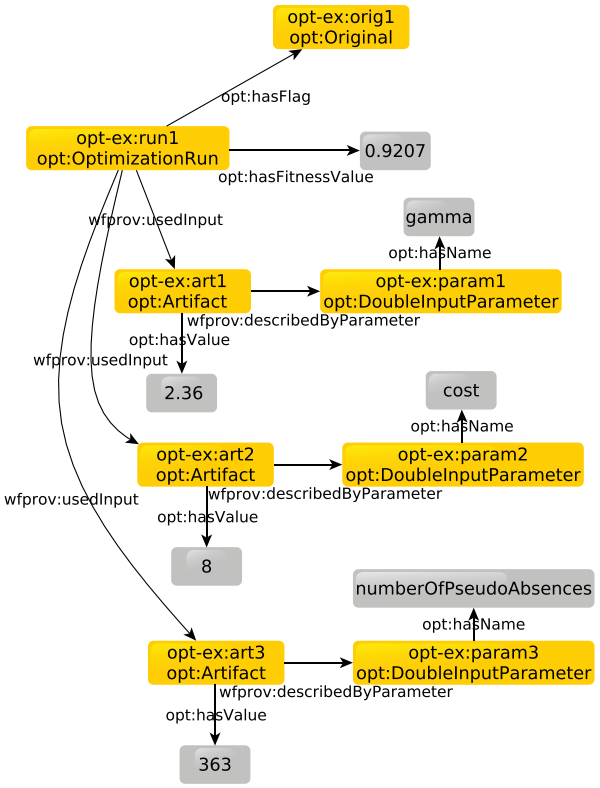

Figure 10: Modeling an Optimization run with the RO-Opt ontology.

version of myExperiment[14] is currently being extended to enable users to create and manage workflow-centric Research Objects. Once stable, the new functionalities will be then incorporated within the production version of myExperiment[15].

One can envisage storing and sharing Optimization Research Objects using a community repository. For example, having the full example of the extract shown in Figure 10 stored in a SPARQL endpoint would easily enable answering a query to retrieve the parameter names (`?pname`) and values (`?value`) of the best optimization runs (i.e., those runs (`?run`) with best fitness values):

```
prefix opt: <http://purl.org/net/RO-optimization#>
prefix wfprov: <http://purl.org/wf4ever/wfprov#>
select distinct ?run ?pname ?value where {
  ?run a opt:OptimizationRun.
  ?run opt:hasFitnessValue ?v.
  ?run wfprov:usedInput ?in.
  ?in opt:hasValue ?value
  ?in wfprov:describedByParameter ?p.
  ?p opt:hasName ?pname.
}order by desc (?v)
```

Other interesting questions would include retrieving the parameters involved as part of the optimization, knowing which is the best parameter combination for a given workflow, gathering the parameter ranges used during prior workflow optimization runs, collecting which was the best result ever gained for a specific workflow or retrieving those parameters which seem to have a larger influence on the final result. By implementing a close connection between workflow management systems (in portals) and infrastructure, optimization provenance could be directly uploaded and downloaded for sharing purposes and further reuse. Researchers could not only retrieve answers through a portal (e.g. myExperiment), but also let optimization algorithms reuse the optimization provenance in many different ways, such as:

- Adopt the used parameter ranges of parameters
- Reuse the algorithm settings
- Perform the identical optimization with different input data
- Use self-defined optimization settings but reuse existing results if possible
- Resume optimization and use half of parameters as half of the initial population of a new optimization run

The more search space definitions, results and other optimization meta-data are stored, the easier it is to determine what the best relevant ranges, parameters, or algorithm settings are. Certainly, similar results could also be obtained by analyzing ordinary workflow runs but Research Objects of optimized workflows bundle important information in a machine readable representation. This concept, implemented by the proposed workflow optimization provenance ontology, may lead to a simplified and automated way to obtain better scientific results. In our ongoing work, we will investigate the possibility to incorporate Optimization Research Objects, as presented in this paper, in a community portal such as myExperiment.

7. CONCLUSIONS AND FUTURE WORK

We presented in this paper the RO-Opt ontology[16], the first proposal for capturing the results of scientific workflow optimizations in a systematic and structured manner. The RO-Opt ontology captures different facets of workflow optimizations, including the workflow (or sub-workflow) subject to optimization, the space of solutions that was explored, the algorithm used for optimization as well as the fitness function used for assessing the fitness of potential solutions. In order to promote its adoption we built RO-Opt upon the Research Object model, a model that has been developed to enable the sharing, reuse and dissemination of scientific research results. We also showcased the use of RO-Opt to encode the optimization results of a real-world scientific workflow and made it available online. By storing optimization provenance, many questions can be answered and meta-data reused, which would not be possible otherwise. The full potential of our approach will be maximized when enabling sharing and reuse in a collaborative manner. Similar to conventional workflow provenance, many researchers will then be able to benefit from the available optimization meta-data.

Our ongoing and future works include promoting the use of RO-Opt in an e-Science infrastructure and investigating further real use cases and with the objective of extending

[14] `http://alpha.myexperiment.org`
[15] `http://www.myexperiment.org`
[16] `http://purl.org/net/RO-optimization#`

it to fit (new) requirements of users and other workflow management systems.

8. ACKNOWLEDGMENTS

The authors would like to thank Alan Williams from the BioVeL project for his support on the DR workflow (BioVeL: EU FP7 283359 BioVeL BioDiversity eLaboratory). This work was partly supported by the myGrid Platform Grant (EPSRC EP/G026238/1, "myGrid: A platform for e-Biology Renewal"), by the Wf4Ever European project (FP7-270192), and an FPU grant (Formacion de Profesorado Universitario) from the Spanish Science and Innovation Ministry (MICINN).

9. ADDITIONAL AUTHORS

Additional authors: Carole Goble (School of Computer Science, University of Manchester, UK,
email: `carole.goble@manchester.ac.uk`).

10. REFERENCES

[1] O. Belhajjame, Khalid andCorcho, D. Garijo, J. Zhao, P. Missier, D. R. Newman, R. Palma, S. Bechhofer, G. C. Esteban, J. M. Gomez-Perez, G. Klyne, K. Page, M. Roos, J. E. Ruiz, S. Soiland-Reyes, L. Verdes-Montenegro, D. De Roure, and C. Goble. Workflow-centric research objects: First class citizens in scholarly discourse. In *In Proceedings of the ESWC2012 Workshop on the Future of Scholarly Communication in the Semantic Web*, 2012.

[2] F. Chirigati, V. Silva, E. Ogasawara, D. de Oliveira, J. Dias, F. Porto, P. Valduriez, and M. Mattoso. Evaluating parameter sweep workflows in high performance computing. In *Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies*, page 2. ACM, 2012.

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

[4] M. E. de Souza Muñoz, R. De Giovanni, M. F. de Siqueira, T. Sutton, P. Brewer, R. S. Pereira, D. A. L. Canhos, and V. P. Canhos. openModeller: a generic approach to species' potential distribution modelling. *GeoInformatica*, 15(1):111–135, 2011.

[5] E. Deelman, D. Gannon, M. Shields, and I. Taylor. Workflows and e-Science: An overview of workflow system features and capabilities. *Future Generation Computer Systems*, 25(5):528–540, 2009.

[6] X. Fan, P. Brézillon, R. Zhang, and L. Li. Making context explicit towards decision support for a flexible scientific workflow system. In *Fourth Workshop on Human Centered Processes*, pages 3–9, 2011.

[7] C. Goble, J. Bhagat, S. Aleksejevs, D. Cruickshank, D. Michaelides, D. Newman, M. Borkum, S. Bechhofer, M. Roos, P. Li, and D. De Roure. myExperiment: a repository and social network for the sharing of bioinformatics workflows. *Nucleic acids research*, 38(suppl 2):W677–W682, 2010.

[8] J. Guinan, C. Brown, M. F. Dolan, and A. J. Grehan. Ecological niche modelling of the distribution of cold-water coral habitat using underwater remote sensing data. *Ecological Informatics*, 4(2):83–92, 2009.

[9] S. Holl, O. Zimmermann, and M. Hofmann-Apitius. A new optimization phase for scientific workflow management systems. In *8th International Conference on E-Science (e-Science)*, pages 1–8. IEEE, 2012.

[10] J. H. Holland. Genetic algorithms. *Scientific american*, 267(1):66–72, 1992.

[11] G. E. Hutchinson. Cold Spring Harbor Symposium on Quantitative Biology. *Concluding remarks*, 22:415–427, 1957.

[12] T. Kiss, P. Greenwell, H. Heindl, G. Terstyanszky, and N. Weingarten. Parameter sweep workflows for modelling carbohydrate recognition. *Journal of Grid Computing*, 8(4):587–601, 2010.

[13] S. A. Kulhanek, B. Leung, and A. Ricciardi. Using ecological niche models to predict the abundance and impact of invasive species: application to the common carp. *Ecological Applications*, 21(1):203–213, 2011.

[14] R. Littauer, K. Ram, B. Ludäscher, W. Michener, and R. Koskela. Trends in Use of Scientific Workflows: Insights from a Public Repository and Recommendations for Best Practice. *International Journal of Digital Curation*, 7(2):92–100, 2012.

[15] B. Ludäscher, I. Altintas, S. Bowers, J. Cummings, T. Critchlow, E. Deelman, D. De Roure, J. Freire, C. Goble, M. Jones, S. Klasky, T. McPhillips, N. Podhorszki, C. Silva, I. Taylor, and M. A. Vouk. Scientific process automation and workflow management. In *Scientific Data Management: Challenges, Technology, and Deployment*, Computational Science Series, chapter 13, pages 476–508. Chapman & Hall, 2009.

[16] K. Miettinen. *Nonlinear multiobjective optimization*. Springer, 1999.

[17] P. Missier, B. Ludäscher, S. Dey, M. Wang, T. McPhillips, S. Bowers, M. Agun, and I. Altintas. Golden trail: Retrieving the data history that matters from a comprehensive provenance repository. *International Journal of Digital Curation*, 7(1):139–150, 2012.

[18] P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. Goble. Taverna, reloaded. In *Proceedings of the 22nd international conference on Scientific and statistical database management*, pages 471–481. Springer, 2010.

[19] S. Smanchat, M. Indrawan, S. Ling, C. Enticott, and D. Abramson. Scheduling parameter sweep workflow in the Grid based on resource competition. *Future Generation Computer Systems*, 29(5):1164–1183, 2013.

[20] J. Stoyanovich, B. Taskar, and S. Davidson. Exploring repositories of scientific workflows. In *Proceedings of the 1st International Workshop on Workflow Approaches to New Data-centric Science*, page 7. ACM, 2010.

[21] T. Weise, M. Zapf, R. Chiong, and A. J. Nebro. Why is optimization difficult? In *Nature-Inspired Algorithms for Optimisation*, volume 193 of *Studies in Computational Intelligence*, pages 1–50. Springer, 2009.

[22] E. O. Wiley, K. M. McNyset, A. T. Peterson, C. R. Robins, and A. M. Stewart. Niche modeling and geographic range predictions in the marine environment using a machine-learning algorithm. *Oceanography*, 16(3):120–127, 2003.