A New Optimization Phase for Scientific Workflow Management Systems

Sonja Holl*, Olav Zimmermann*, Martin Hofmann-Apitius+

*Jülich Supercomputing Centre (JSC)
Institute for Advanced Simulation (IAS)
Forschungszentrum Juelich, Germany

+Fraunhofer Institute for Algorithms
and Scientific Computing SCAI
Schloss Birlinghoven, Sankt Augustin, Germany

8th IEEE International Conference on eScience 2012 - Chicago
Scientific Workflows

- Popular choice to realize *in silico* experiments without programming
- Define data flow between scientific components
- Model, design, share and execute scientific services

Taken from: www.myexperiment.co.uk, Paul Fisher, Title: Pathways and Gene annotations for Arabidopsis affy data
Designing workflows (from scratch)

Reusing workflows

in silico experiment

Execution

Sharing

Output

Data

Data

Inspired by Sarah Cohen-Boulakia
Designing workflows (from scratch)

Reusing workflows

in silico experiment

Execution

Data

Output

Execution

Data

There is more…

Inspired by Sarah Cohen-Boulakia
Inspired by Sarah Cohen-Boulakia

Designing workflows (from scratch)

Reusing workflows

Sharing

Execution

Data

There is more…

BLAST®
Inspired by Sarah Cohen-Boulakia

- Designing workflows (from scratch)
- Reusing workflows
- Sharing data
- Execution

There is more…
Why? Scientific Workflow Optimization

Common strategies:
  - using default parameters
  - keep known application

But: Typically results do not match expectations

Improving workflow results by:
  - reusing previous parameter settings
  - trial and error
  - parameter sweeps
How? Scientific Workflow Optimization

But:

- Increasing complexity of scientific workflows
- Raising number of configuration parameter and available algorithms and applications
- Manual optimization techniques are inefficient (trial and error, parameter sweep)
How? Scientific Workflow Optimization

But:
- Increasing complexity of scientific workflows
- Raising number of configuration parameter and available algorithms and applications
- Manual optimization techniques are inefficient (trial and error, parameter sweep)

Our Approach:
- Intelligent automated optimization techniques
Outline

- Motivation
- Scientific Workflow Optimization
  - Extending the Scientific Workflow Life Cycle
  - Workflow Optimization Types
- Implementation
  - API and Plugin Architecture
  - Example: Parameter Optimization via GA
- Bioinformatics Use Case
- Future Work & Conclusion
The Scientific Workflow Life Cycle in Literature

Optimization in the Scientific Workflow Life Cycle

- Design and Refinement
- Analysis and Learning
- Planning and Sharing
- Execution
- Optimization

Optimization flows back to Planning and Sharing, which in turn feeds into Analysis and Learning, ultimately leading back to Design and Refinement.
Scientific Workflow Optimization - 1

Parameter Optimization

- Find optimal settings for the scientific workflow
- Find setting that optimizes output
- Use heuristic search algorithm
- Use fitness value for evaluation
Extended Parameter Optimization

- Find optimal input parameter for WF components

- Integrate users knowledge and component specific limits:
  - Value limits (1>a>8)
  - Pairwise dependent parameters
    - Mathematical (a+b=2, a+b<100)
    - Logical ( a v b)
  - Fixed value combinations ( a=8/b=2 or a=9/b=1 or a=9/b=2 )
Scientific Workflow Optimization - 3

Component Optimization

- Find optimal components for the scientific workflow

- Several methods available for same functionality
- Replace component by an equivalent one
- “Equivalence“ described by the user or by an ontology
- Minimum Requirement: same input/output signature
- Replace equivalent sub-workflows
- Example: Different implementations of protein sequence matching (Smith-Waterman, BLAST etc.)
Scientific Workflow Optimization - 4

**Composition Optimization**

- Find optimal composition for the scientific workflow
- Reorganization of the data or enactment flow
- Swapping of workflow components or sub-workflows
- Minimum Requirement: same input/output signature
- Examples: hierarchical SVM multiclass classification, multi step filtering/clustering
Provenance based Optimization

Idea: Learning from prior workflow optimizations

Requirements:
- Repository of provenance data and optimization results (including intermediate results)
- Sharing through eScience infrastructure
- Adaptable for all workflow optimization types
- Guided optimization processes
Scientific Workflow Optimization API

- Integrate abstract optimization phase into workflow management systems
- Interrupt the workflow life cycle BEFORE execution
- Typically only sub-workflows need optimization
- Optimize the sub-workflow(s) independently
- API for parameter, component, and composition optimization
Taverna Scientific Workflow Management System

- Environment for scientific workflows
- Automate execution, data handling, monitoring, scheduling of workflows
- Open source written in Java
- Components represent Web Services, Java classes, local scripts, Grid Services or others
Taverna Optimization API Implementation

- Taverna workflow management system
- Implementation of optimization API
- Usage of the Taverna processor dispatch stack
- Optimization plugins can be integrated independently
New Taverna Optimization Layer

- Taverna layers implement different steps for activity invocation
- Optimization layer
  - extracts workflow details
  - calls the corresponding plugin
  - delegates the execution of the workflows
  - collects results

<table>
<thead>
<tr>
<th>Optimization Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimize</td>
</tr>
<tr>
<td>Parallelize</td>
</tr>
<tr>
<td>Error bounce</td>
</tr>
<tr>
<td>Failover</td>
</tr>
<tr>
<td>Retry</td>
</tr>
<tr>
<td>Invoke</td>
</tr>
</tbody>
</table>
Example Plugin: Parameter Optimization

- Using Genetic Algorithm for heuristic search
- Representing parameters as genome

- Using established GA Framework: JGAP
  - special MapGene for fixed parameter values
  - encoding of limits and spacing for numerical parameters
  - encoding of parameter dependencies
  - adjusted the main Breeder method
Taverna Workflow Optimization

- Graphical User Interface for optimization settings
Use Case: Support Vector Regression

- Estimate local structural similarity of a protein segment to a reference segment with defined secondary structure
- Goal: training a set of estimators to predict secondary structure
- Fitness function: Squared correlation coefficient between predicted and observed structure distances (optimum: 1)

- Parameters for optimization:
  - $\gamma$: width of Gaussian for RBF kernel
  - $\varepsilon$: parameter for error penalization
  - $C$: penalizes complexity of the model
Use Case: Support Vector Regression - Results

Manually guided parameter sweep using 420 samples

- Optimum: 0.527902
- With C=64, γ=0.03125, ε=0.75

Using GA Optimization Plugin in Taverna with 100 samples

- Optimum: 0.588481
- With C=61, γ=0.03342, ε=1.5758

- Constraints C: [2^0; 2^8], γ : [2^{-12}; 2^0], ε: [0.0; 10.0].
Outlook

- Integrating component optimization into Taverna
- Demonstrate performance with other use cases
- Extending the idea of workflow optimization in eScience infrastructures
- User input for fitness value estimation
Conclusion

- New optimization phase to assist researchers obtaining better results
- Definition of different workflow optimization types
- API for different optimization Plugins
- Implementation of API and GUI in Taverna
- Example: GA-based parameter optimization
  - Optimization required 4x fewer evaluations than parameter sweep

Many thanks to the Taverna Team
Questions?
Thank you!