Rewriting scientific workflows while preserving provenance

Sarah Cohen-Boulakia  
Christine Froidevaux  
Jiuqiang Chen  

Université Paris Sud  
LRI CNRS UMR 8623, INRIA
Operations on workflows

Designing workflows (from scratch)

Clustering (sub)graphs

Graphs mapping

Graphs matching (subgraph isomorphism)

Querying workflows

Storing (indexing) executions

Comparing workflows (or executions)

Comparing graphs

The cost of these operations is highly dependent on the workflow structures

in silico experiment

Sarah Cohen-Boulakia, Univ. Paris-Sud, eScience, October 11th 2012
Structure of scientific workflows: DAGs

- Workflow example from Taverna
- Specification
- Execution
These operations lead to NP-hard problems when DAGs are considered!
Series-Parallel graphs

- Introduced by Duffin to model electrical networks

- Simple but expressive graph structures
  - Capture the structure of most business workflows
  - Have been used for representing scientific workflows

- Complex operations have linear time solutions
  - SubGraph isomorphism
  - Graph comparision
  - Some kinds of graph clustering operations

→ Linear time algorithms can be designed to enhance workflow design, exchange, querying, comparision...
Outline

1. Motivation
2. *Series-Parallel workflows*
3. Provenance equivalence
4. Transformation Algorithm
5. Experiments
6. Conclusions and Discussions
Definition of SP-graphs (1/2)

**G is SP iff MaxRed(G) = BSP**

- **MaxRed(G):** iteratively performs series and parallel reductions on a given graph G

- **BSP:** Basic Series-Parallel
Paral red \((s,1)\)

3 Series red \((2,3)(3,6)\)
\((2,4)(4,6)\)
\((2,5)(5,6)\)

Series red \((s,1)\) \((1,2)\)

Paral red \((2,6)\)

Series red \((2,6)(6,t)\)

Paral red \((2,t)\)

4 Series red \((6,7)(7,8)\)
\((7,8)(8,9)\)
\((8,9)(9,10)\)
\((9,10)(10,11)\)

\((G_0)\)
Another definition (Non SP-graphs)

$G$ is non-SP iff $\text{MaxRed}(G)$ contains $G_{\text{forbidden}}$

$G_{\text{forbidden}}$

\[ v \quad w \]

$v$ and $w$ are called reduction nodes

Intuitively, such graphs cannot be synchronized

→ Approaches are available to transform Non SP graphs into SP graphs

• Do they preserve provenance?
Outline

1. Motivation
2. Series-Parallel workflows
3. Provenance equivalence
4. Transformation Algorithm
5. Experiments
6. Conclusions and Discussions
# Provenance model

- **Immediate** provenance
  \[
  \text{imProv}(f)_{\text{Grun}} = u \cdot (d_1 + \cdots + d_p)
  \]

- **Deep** provenance
  \[
  \text{DProv}(f)_{\text{Grun}} = u \cdot (d_1 \cdot \text{DProv}(e_1)_{\text{Grun}} + \cdots + d_p \cdot \text{DProv}(e_p)_{\text{Grun}})
  \]
  \[
  \text{imProv}(f)_{\text{Grun}} = \text{DProv}(f)_{\text{Grun}} = s
  \]

- **Out Provenance**
  \[
  \text{outProv}(G_r) = d_1 \cdot \text{DProv}(e_1)_{\text{Grun}} \cup \cdots \cup d_n \cdot \text{DProv}(e_n)_{\text{Grun}}
  \]

---

**Example**

- \(e_1(d_1)\)
- \(e_2(d_2)\)
- \(e_3(d_3)\)
- \(e_4(d_4)\)
- \(e_5(d_5)\)

\[\text{imProv}(e_4)_{G_r} = v \cdot (d_1 + d_3)\]
\[\text{DProv}(e_4)_{G_r} = v \cdot (d_1 \cdot s + d_3 \cdot u \cdot d_2 \cdot s)\]
\[\text{outProv}(G_r) = \text{DProv}(e_4)_{G_r} \cup \text{DProv}(e_5)_{G_r}\]
**Provenance equivalent transformations?**

G1 and G2 are **provenance-equivalent** iff
\[ \text{outProv}(G_1) = \text{outProv}(G_2) \]

- Rewriting approaches are not provenance equivalent
  - **Example:** Synchronization techniques [Escribano et al.]

**Synchronizations do not preserve provenance!**

Forbidden graph

In G2 d5 depends on d4 which is not the case in G1
\[ \text{outprov}(G1) \neq \text{outprov}(G2) \]
Our proposal

- Node duplication based on outputs
- Does not change the behavior of the task

The two graphs are provenance-equivalent

Node duplication preserves provenance
Outline

1. Motivation
2. Series-Parallel workflows
3. Provenance equivalence
4. Transformation Algorithm
5. Experiments
6. Conclusions and Discussions
SPFlow algorithm

Initialization
1. SPG ← G; s ← s(G); t ← t(G)
2. G_{red} ← \text{MaxRed}(G)
3. Find autonomous subdags in G_{red}
4. TransfoRec(G,G_{red}, SPG, s, t)

TransfoRec (IN: G; IN/OUT: G_{red}; IN/OUT: SPG; IN: s, t)
Repeat until G_{red} is BSP
  step 1: Start from s in G_{red}, find a reduction node x
  step 2: if x is the source of a autonomous subdag G[v,w],
      with x=v then TransfoRec(G,G_{red}, SPG, v, w)
      else
      1. Perform node reduction of x on G_{red}
      2. Perform node duplication of x on SPG
      3. G_{red} ← \text{MaxRed}(G_{red})
Example

$G$ $\rightarrow$ $G_{red}$ $\rightarrow$ MaxRed($G_{red}$)

Node $red(u)$

$G_{red} = BSP$

Node $dupl(u)$

$SPG$ $\rightarrow$ $SPG$
Outline

1. Motivation
2. Series-Parallel workflows
3. Provenance equivalence
4. Transformation Algorithm
5. Experiments
6. Conclusions and Discussions
Structural features of scientific workflows in myExperiment

- Evolution of SP structures (Taverna workflows)

<table>
<thead>
<tr>
<th>Data</th>
<th>Number of workflows</th>
<th>SP graphs (proportion)</th>
<th>Non SP graphs (proportion)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>681</td>
<td>429 (63%)</td>
<td>252 (37%)</td>
</tr>
<tr>
<td>2011</td>
<td>879</td>
<td>554 (63%)</td>
<td>325 (37%)</td>
</tr>
<tr>
<td>2012</td>
<td>1014</td>
<td>624 (61.5%)</td>
<td>390 (38.5%)</td>
</tr>
</tbody>
</table>

- SP Structures in families (may 2012)

<table>
<thead>
<tr>
<th>Family</th>
<th>#workflows</th>
<th>%Non-SP structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple(1-3 nodes)</td>
<td>92</td>
<td>0%</td>
</tr>
<tr>
<td>Medium(4-10 nodes)</td>
<td>480</td>
<td>13%</td>
</tr>
<tr>
<td>Complex(11-20 nodes)</td>
<td>214</td>
<td>55%</td>
</tr>
<tr>
<td>very complex(≥ 21 nodes)</td>
<td>228</td>
<td>93.3%</td>
</tr>
</tbody>
</table>
Number of reduction nodes

- The number of reduction nodes provides a measure from Non SP to SP

- 1 reduction node → 31%

- 1-3 reduction nodes → 67%
Most of the workflows < 4

Worst result: >13 in a very few cases

→ Linear!
Operations on workflows (SP)

- Designing workflows
- Storing (indexing) executions
- Scheduling executions
- Comparing workflows (or executions)
- Comparing graphs
- Graphs matching (isomorphism)
- Clustering (sub)graphs
- Polynomials
- Polynomial
- Polynomial
- Querying workflows

Sarah Cohen-Boulakia, Univ. Paris-Sud, eScience, October 11th 2012
Contributions

1. Proposition of a **model** to represent scientific workflows and provenance generated in their execution

2. Definition of the notion of **provenance-equivalence**

3. Review of several rewriting strategies and proof that they are not provenance-equivalent

4. Design of the provenance-equivalent **SPFlow** algorithm to translate Non-SP workflows into SP workflows

5. **Evaluation** of our approach on a thousand of scientific workflows
Ongoing work

- Provide a web service to rewrite Taverna workflows
  - Evaluating the cost of duplication (running rewritten workflows vs running original workflows)

- Understand the causes of non SPness
  - Collaborative work with C. Goble and A. Williams (University of Manchester)
  - Focus on a set of workflows from the BioVel project

- Provide guidelines to users to design SP workflows
  -> Distilling workflow structure

- Extend our work on more complex workflow structures and other systems
  - SPFL (Fork-Loop)
  - Kepler, VisTrails, ...
Acknowledgements

Members of the AMIB/bioinfo group (Ch. Froidevaux)
Carole Goble
Paolo Missier
Alan R. Williams
Khalid Belhajjame

Members of the Penn DB group
Ulf Leser
Bertram Ludäscher
Shawn Bower
Juliana Freire