Putting Lipstick on Pig: Enabling Database-style Workflow Provenance

Yael Amsterdamer, Susan B. Davidson, Daniel Deutch Tova Milo, Julia Stoyanovich, Val Tannen

Presented by Guozhang Wang

DB Lunch, Apr. 23rd, 2012
A Story of “How Research Ideas Get Motivated”

- A short time ago, somewhere in the Globe of CS Research …
Workflow Provenance

- Motivated by Scientific Workflows

- Community: IPAW
- Interests: process documentation, data derivation and annotation, etc.
- Model: OPM
OPM Model

- Annotated directed acyclic graph
  - Artifact: immutable piece of state
  - Process: actions performed on artifacts, result in new artifacts
  - Agents: execute and control processes

- Aims to capture causal dependencies between agents/processes

- Each process is treated as a “black-box”
Meanwhile

- On the other side of the Globe …
Data Provenance (for Relational DB and XML)

- Motivated by Prob. DB, data warehousing ..

- Community: SIGMOD/PODS

- Interests: data auditing, data sharing, etc

- Model: Semiring (etc)
Semiring

- **K-relations**
  - Each tuple is uniquely labeled with a provenance “token”

- **Operations:**
  - • : join
  - + : projection
  - 0 and 1: selection predicates
A Datalog Example of Semiring

\[ q(x,z) :\text{-} R(x,_,z), \ R(_,_,z) \]
\[ q(x,z) :\text{-} R(x,y,_), \ R(_,y,z) \]

\[
\begin{array}{ccc|c}
  a & b & c & p \\
  d & b & e & r \\
  f & g & e & s
\end{array}
\]

\[
\begin{array}{ccc|c}
  a & c & 2p^2 \\
  a & e & \text{pr} \\
  d & c & \text{pr} \\
  d & e & 2r^2 + rs \\
  f & e & 2s^2 + rs
\end{array}
\]

Slide borrowed from Green et al.
They Live Happily and Semi-Separately, Until …

Data Provenance Researchers

Workflow Provenance Researchers
Semiring Comes to Meet OPM
OPM’s Drawbacks in Semiring People’s Eyes

- The black-box assumption: each output of the module depends *solely* on *all* its inputs
  - Cannot leverage the common fact that some output only depends on small subset of inputs
  - Does not capture internal state of a module

- So: replace it with Semirings!
The Idea

- General workflow modules is complicated, and thus hard to capture its internal logic by annotations.

- However, modules written in Pig Latin is very similar to Nested Relational Calculus (NRC), thus are much more feasible.

- Let us write a paper, wohoh!
End-of-Story Disclaimer

This story is purely imaginative.

It is to be coincidental if there are similarities between the story and the real world.
Pig Latin

- **Data:** unordered (nested) bag of tuples

- **Operators:**
  - FOREACH \( t \) GENERATE \( f_1, f_2, \ldots \) OP(\( f_0 \))
  - FILTER BY condition
  - GROUP/COGROUP
  - UNION, JOIN, FLATTEN, DISTINCT …
Example: Car Dealership.
Bid Request Handling in Pig Latin

\[
\begin{align*}
\text{ReqModel} & = \text{FOREACH Requests GENERATE Model;} \\
\text{Inventory} & = \text{JOIN Cars BY Model, ReqModel BY Model;} \\
\text{SoldInventory} & = \text{JOIN Inventory BY CarId, SoldCars BY CarId;} \\
\text{CarsByModel} & = \text{GROUP Inventory BY Model;} \\
\text{SoldByModel} & = \text{GROUP SoldInventory BY Model;} \\
\text{NumCarsByModel} & = \text{FOREACH CarsByModel GENERATE group as Model, COUNT(Inventory) as NumAvail;} \\
\text{NumSoldByModel} & = \text{FOREACH SoldByModel GENERATE group as Model, COUNT(SoldInventory) as NumSold;} \\
\text{AllInfoByModel} & = \text{COGROUP Requests BY Model, NumCarsByModel BY Model, NumSoldByModel BY Model;} \\
\text{InventoryBids} & = \text{FOREACH AllInfoByModel GENERATE FLATTEN(CalcBid(Requests, NumCarsByModel, NumSoldByModel))};
\end{align*}
\]
Provenance Annotation
Provenance Annotation 1.1

- Provenance node and value nodes
  - Workflow input nodes
  - Module invocation nodes
  - Module input/output nodes
Provenance Annotation 1.2

- State nodes
  - P-node for the tuple
  - P-node for the state
Provenance Annotation 2.1

- FOREACH (projection, no OP)
  - P-node with “+”
Provenance Annotation 2.2

- JOIN
  - P-node with “⋆”

![Diagram showing JOIN operation with P-node marked by “⋆” and corresponding tables with CarId and Model information.](image)
Provenance Annotation 2.3

- **GROUP**
  - P-node with “∂”

<table>
<thead>
<tr>
<th>Requests</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>UserId</td>
<td>BidId</td>
</tr>
<tr>
<td>$P_1$</td>
<td>$B_1$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Cars</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CarId</td>
<td>Model</td>
</tr>
<tr>
<td>$C_1$</td>
<td><em>Accord</em></td>
</tr>
<tr>
<td>$C_2$</td>
<td><em>Civic</em></td>
</tr>
<tr>
<td>$C_3$</td>
<td><em>Civic</em></td>
</tr>
</tbody>
</table>
Provenance Annotation 2.4

- FOREACH (aggregation, OP)
  - V-node with the OP name
Provenance Annotation 2.5

- COGROUP
  - P-node with “∂”
Provenance Annotation 2.6

- FOREACH (UDF Black Box)
  - P-node/V-node with the UDF name
Query Provenance Graph

- Zoom-In v.s. Zoom-Out

Coarse-grained

Fine-grained
Query Provenance Graph

- Deletion Propagation
  - Delete the tuple P-node and its out-edges
  - Repeated delete P-nodes if
    - All its in-edges are deleted
    - It has label • and one of its in-edges is deleted
Implementation and Experiments

- Lipstick prototype
  - Provenance annotation coded in Pig Latin, with the graph written to files
  - Query processing coded in Java and runs in memory.

- Benchmark data
  - Car dealership: fixed workflow and # dealers
  - Arctic Station: Varied workflow structure and size
Annotation Overhead

- Overhead increases with execution time
Annotation Overhead

(c) Car dealerships, impact of parallelism

- Parallelism helps with up to # modules
Loading Graph Overhead

- Increase with graph size (comp. time < 4 sec)
Loading Graph Overhead

Feasible with various sizes (comp. time ~ 8 sec)
Subgraph Query Time

- Query efficiently with sub-second time
Conclusions

• Data provenance ideas such as Semirings can be brought to workflow provenance for those “relational” programs

• No second conclusion, sorry ..

Thank You!
Backup Slides
The introduction of MapReduce/Dryad/Hadoop …
  ◦ Originally designed for data-driven web applications
  ◦ Helped gaining DB researchers attentions back to workflow apps