Policy-Based Integration of Provenance Metadata

Ashish Gehani    Dawood Tariq    Basim Baig
SRI International

Tanu Malik
University of Chicago
Background

- Traditional science:
  - Human hypotheses
  - Experimental, theoretical verification

- Computational paradigm:
  - Automated exploration
  - Machine verification

- Provenance tracking need:
  - Reproducibility (execution context)
  - Data sharing (dependencies)
  - Scenario analysis (application profiles)
Open Provenance Model

- **Vertex types**
  - Artifact
  - Process
  - Agent

- **Edge types**
  - Used
  - WasGeneratedBy
  - WasDerivedFrom
  - WasControlledBy
  - WasTriggeredBy

- Domain semantics in annotations
Integration Need

- Genome analysis provenance record

![Diagram showing integration need with nodes and arrows representing data flow and integration points.]

- NCBI
- JGI
- TIGR
- PDB
- Swiss-Prot
- Data Ingestion
- GADU Server
- Pegasus Planner
- Comparative Analysis
- Database

Legend:
- Curated provenance
- Application provenance
- Workflow provenance
- Operating system provenance

300 Nodes
Integration Issues

- Metadata variation:
  - Abstraction levels
  - Completeness
  - Identifiers
  - Semantics

- Querying requires:
  - Record assembly
  - Reconciling syntax
  - Mapping semantics
Provenance Middleware

- Support for Provenance Auditing in Distributed Environments
Implementing Policy

- Annotations are key-value pairs
- Filters operate on vertex / edge stream
- Arbitrary transformations possible
Sample Workload

- BLAST (sequence alignment)
- Influenza data
  

- Database construction
  
  makeblastdb -in influenza.faa
  -parse_seqids -hash_index -out outputdb
## Aggregation Policies

<table>
<thead>
<tr>
<th>Policy</th>
<th>Version created after:</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>Every write</td>
</tr>
<tr>
<td>SEQ</td>
<td>Each sequence of <code>read()</code> or <code>write()</code></td>
</tr>
<tr>
<td>CA</td>
<td>Cycle avoidance (Harvard)</td>
</tr>
<tr>
<td>GF</td>
<td>Graph finesse (Harvard)</td>
</tr>
<tr>
<td>OC</td>
<td><code>open()</code> - <code>close()</code> pair</td>
</tr>
</tbody>
</table>
Size of different *aggregation policy* filters.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Lines of code</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL</td>
<td>38</td>
</tr>
<tr>
<td>SEQ</td>
<td>73</td>
</tr>
<tr>
<td>CA</td>
<td>63</td>
</tr>
<tr>
<td>GF</td>
<td>80</td>
</tr>
<tr>
<td>OC</td>
<td>6</td>
</tr>
</tbody>
</table>
Artifact Vertices

Effect of *aggregation policies* over time.

![Graph showing effect of aggregation policies over time. The graph plots the number of vertices created against time (s), with lines for different aggregation policies: ALL, SEQ, CA, GF, OC.](image)
Used Edges

- Effect of *aggregation policies* over time.

![Graph showing the effect of aggregation policies over time. The graph compares different aggregation policies with lines for ALL, SEQ, CA, GF, and OC.]
Effect of *aggregation policies* over time.

![Graph showing the effect of aggregation policies over time.](image)
Conclusion

- Provenance integration policy matters
- Substantial impact:
  - Runtime - CPU, memory
  - Persistent - storage, querying
- Acknowledgement
  - US NSF Grant OCI-0722068
- URL: http://spade.csl.sri.com
- Code license: GPLv3
- Email: ashish.gehani@sri.com
- Questions?