Study in Usefulness of Middleware-Only Provenance

Quan(Gabriel) Zhou, Devarshi Ghoshal, Beth Plale

May 6, 2016

DATA TO INSIGHT CENTER
INDIANA UNIVERSITY
Pervasive Technology Institute

Main Contents

- I. Contributions
- II. Overview
- III. Framework
- IV. Completeness of middleware provenance
- V. Evaluation
- VI. Discussion & Limitation
- VII. Conclusion & Future Work

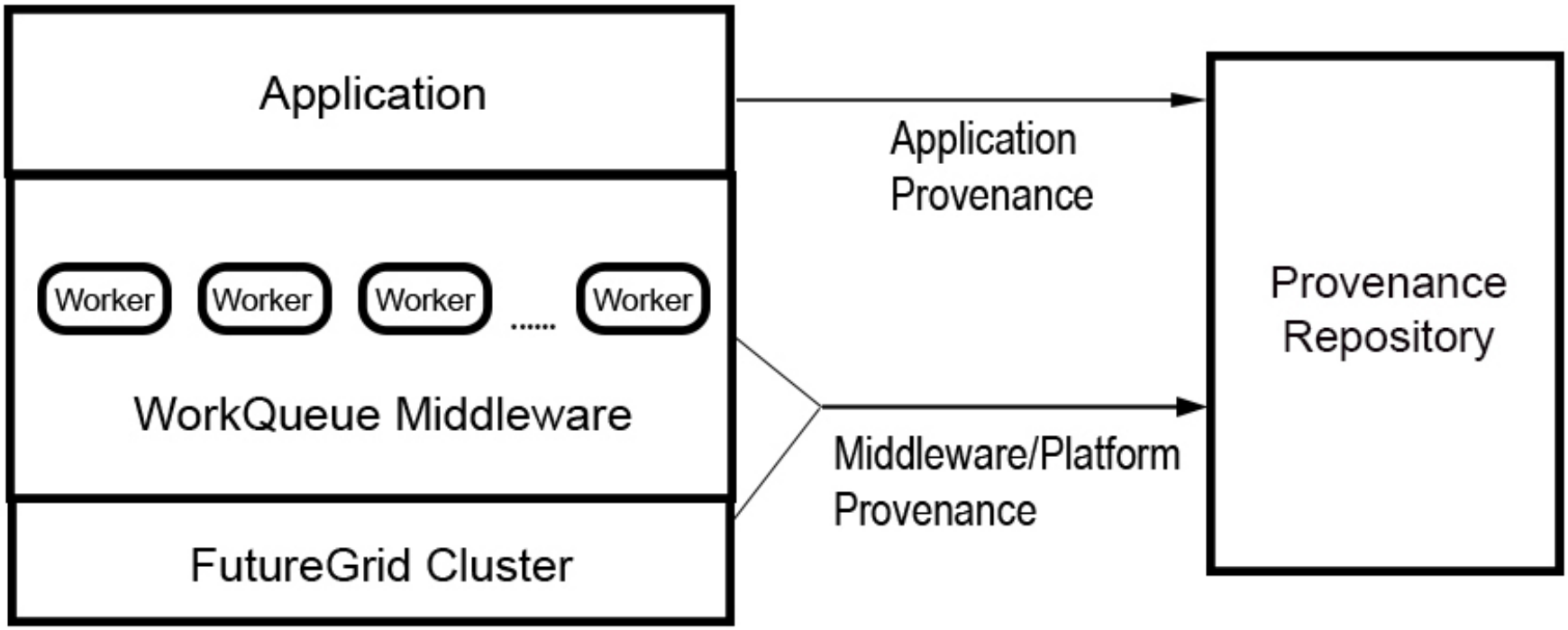
Contributions

I. A provenance collection extension to WorkQueue that requires no application instrumentation and minimal system intrusion

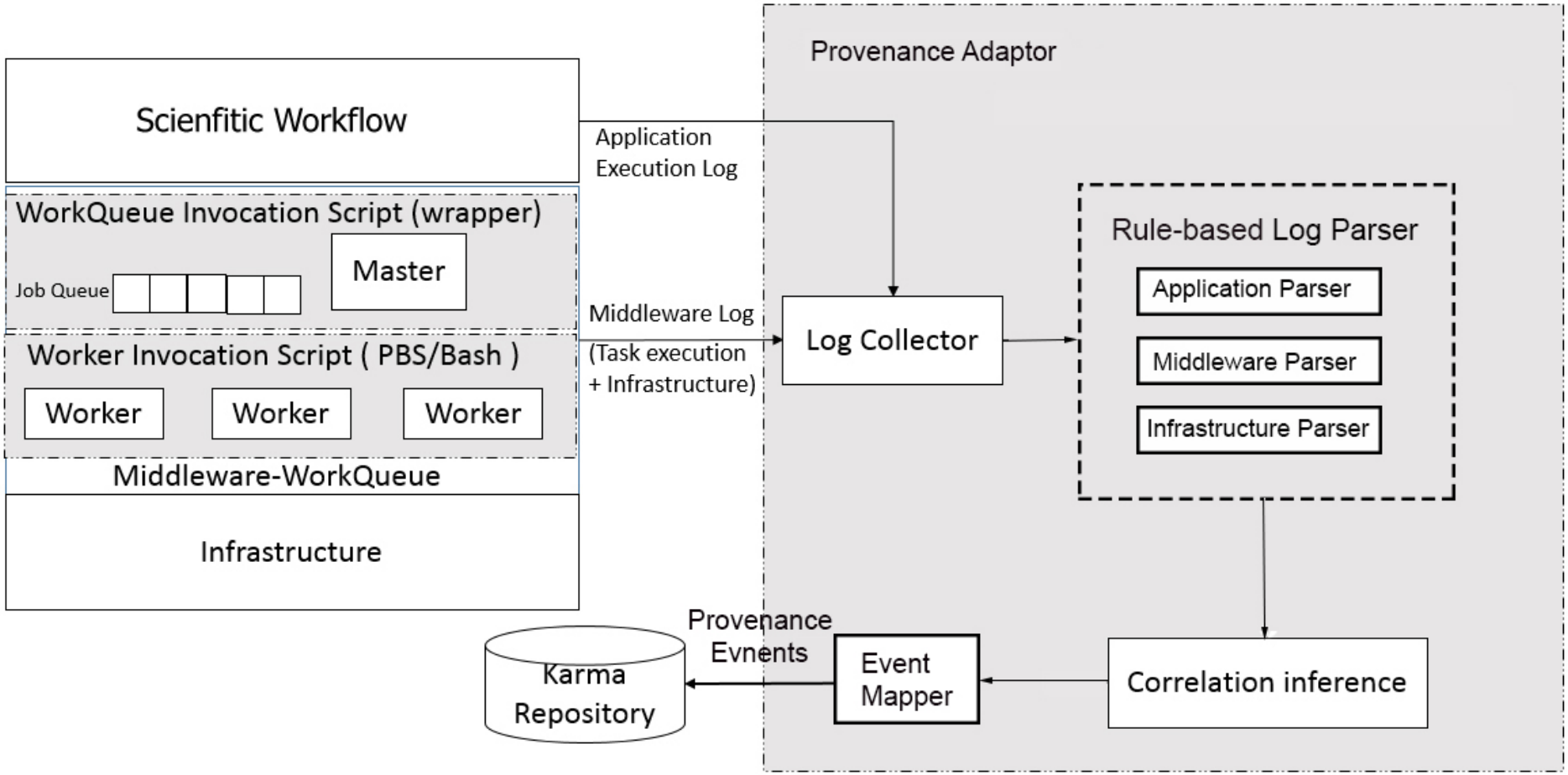
II. A technique to correlate provenance events from different layers

III. An evaluation to show the completeness of middleware layer provenance and usefulness in failure tracing and classification

Overview


Application
Worker
Worker
Worker
.....
Worker
WorkQueue Middleware
FutureGrid Cluster
Application
Provenance
Middleware/Platform
Provenance
Provenance
Repository

Framework


Provenance Adaptor
Scienfitic Workflow
Application Execution Log
WorkQueue Invocation Script (wrapper)
Job Queue
Master
Middleware Log (Task execution + Infrastructure)
Log Collector
Rule-based Log Parser
Application Parser
Middleware Parser
Infrastructure Parser
Worker Invocation Script (PBS/Bash)
Worker
Worker
Worker
Middleware-WorkQueue
Infrastructure
Provenance Evnents
Karma Repository
Event Mapper
Correlation inference

Provenance-aware Middleware

A). A provenance-enabled wrapper

- Set up master/worker environment
- Transparently capture the provenance trace in executing a distributed application through WorkQueue
- Capture information related to failures and data corruption
- “What” and “Why” provenance

B). A worker invocation script

- Configure workers for executing an application
- Provenance captured accounts for task distribution and location of data generated in a distributed environment
- “Where” provenance

Provenance Adaptor

A). Log Collector

B). Log Parser

C). Correlation Inference Engine

D). Event Mapper

Log Collector & Log Parser

a) Middleware layer-specific rule set

b) Application layer-specific rule set

```
<mwp:ruleset>
      <mwp:rule>
          <mwp:keyWord>hostname</mwp:keyWord>
          <mwp:criterion>simple</mwp:criterion>
          <mwp:occurrence>1</mwp:occurrence>
      </mwp:rule>
      <mwp:rule>
         <mwp:startingWord>task
             received</mwp:startingWord>
         <mwp:endingWord>task
             completed</mwp:endingWord>
         <mwp:keyWord>starting
             task</mwp:keyWord>
         <mwp:criterion>complex</mwp:criterion>
         <mwp:occurrence>any</mwp:occurrence>
      </mwp:rule>
</mwp:ruleset>
```

Correlation Inference Engine

-- Presumptions

- Correlate provenance events from different system layers in order to generate complete provenance graphs

- Provenance event tuple presentation:

  (I, O, P, T_s, T_e)

- Synchronized Clock

Correlation Inference Engine

- Data-flow Correlation: (I, O, P)-Analysis
- Timestamp Correlation:

$$E_A(i).T_s \leq E_M(j).T_s \text{ and } E_A(i).T_e \geq E_M(j).T_e \quad (1)$$

$$simT = \frac{|\ B(T_s)\ | + |\ B(T_e)\ |}{T_r} \quad (2)$$

$B(T_s) \leftarrow |\ E_M(i).T_s - E_A(j).T_s\ |$ // Start time bias
$B(T_e) \leftarrow |\ E_M(i).T_e - E_A(j).T_e\ |$ // End time bias
$T_r \leftarrow E_A(j).T_e - E_A(j).T_s$// $E_A(j)$ time range

Event Mapper

- Mapping the raw provenance data to structured provenance events using OPM for provenance representation and exchange

- Set to a provenance management service which internally stores the data as relational tuples for future query and analysis

Completeness of Middleware Provenance

- Contextual analysis for comparing the completeness of provenance from middleware layer to both middleware and application layer
- Equation for completeness index C_M of middleware provenance

$$C_M = \frac{1}{w_e + w_d + w_r + w_a} \times \left(\frac{|E_M| \times w_e}{|E_{M+A}|} + \frac{|D_M| \times w_d}{|D_{M+A}|} + \frac{|R_M| \times w_r}{|R_{M+A}|} + \frac{|A_M| \times w_a}{|A_{M+A}|} \right) \quad (3)$$

Entities(E), Artifacts(D), Relationships(R), Annotations(A), Middleware(M) and Application(A) layer, weights(w_x)

Evaluation

I. Correlation Inference
II. Failure tracing
III. Performance overheads
IV. Completeness evaluation

*** All the tests are performed on a set of bare metal nodes on FutureGrid India clusters each of which is an IBM iDataPlex dx 360 M2 node with Intel Xeon X5550 processor @2.66GHz with 2 CPUs and 8 cores/processor. It has 64-bit Red-hat Enterprise Linux server and 24GB DDR3 RAM. Each execution is done on 4 to 32 node clusters with 8 processors per node.We use Karma V3.2.3 with RabbitMQ V3.2.0 for provenance storage and retrieval.*

Applications

- Atmospheric/oceanographic Workflow: SLOSH application(Sea, Lake, and Overland Surges from Hurricanes)
- Bioinformatics workflows: BLAST, BWA and SHRiMP

TABLE I: Multi-dimensional classification of applications

	SLOSH	BWA	BLAST	SHRiMP
Structure Pattern	Parallel-merge-split	Parallel-merge-split	Parallel-merge-split	Parallel-merge-split
Data Pattern	Data reduction	Data processing	Data reduction	Data production
Usage Scenarios	Event-driven	Interactive	Interactive	Interactive

Correlation Inference

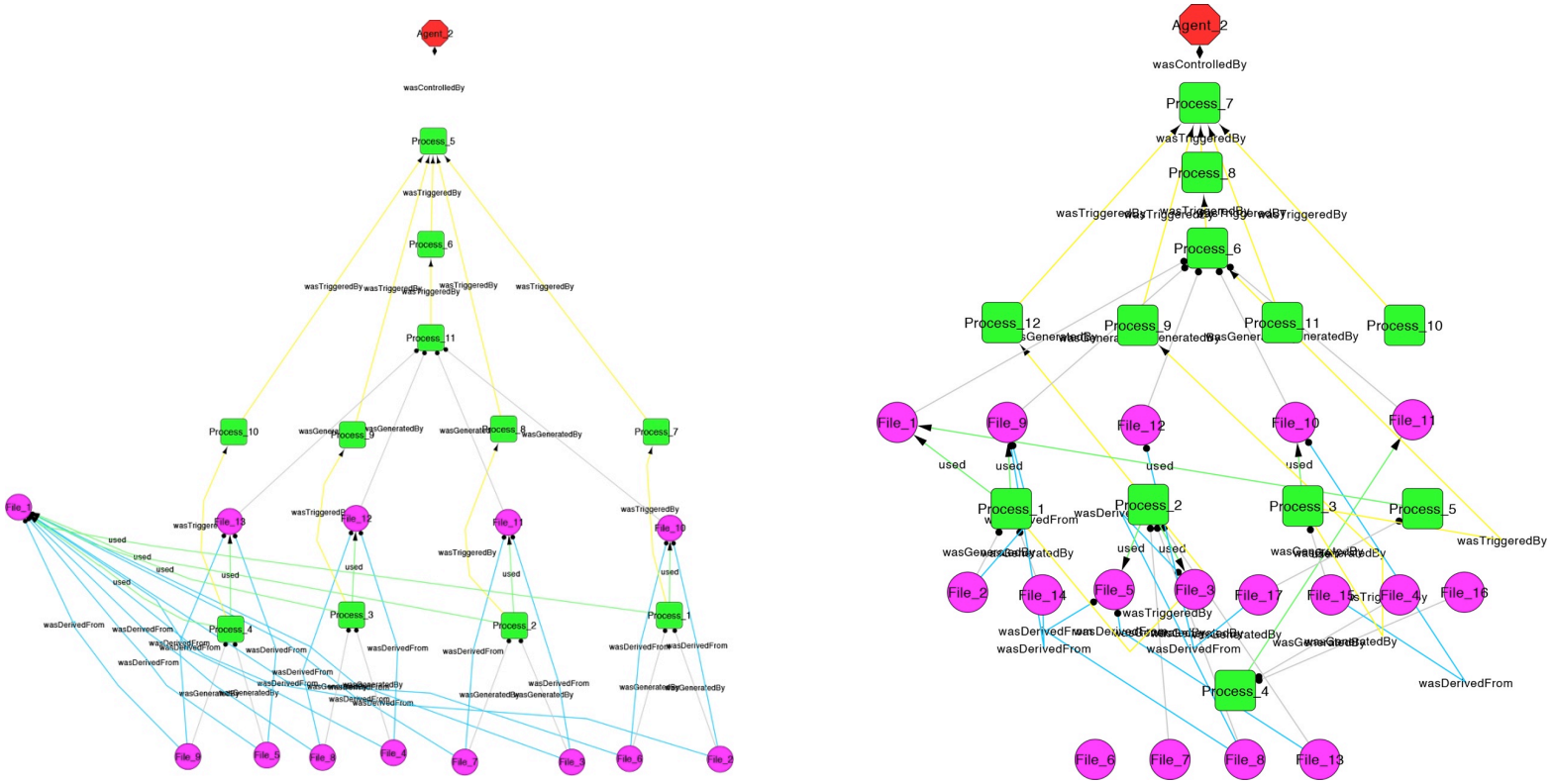


Fig. Layered provenance for SLOSH. The graphs show the significance and extent of middleware provenance as opposed to total provenance which is comprised of middleware and application layer provenance.

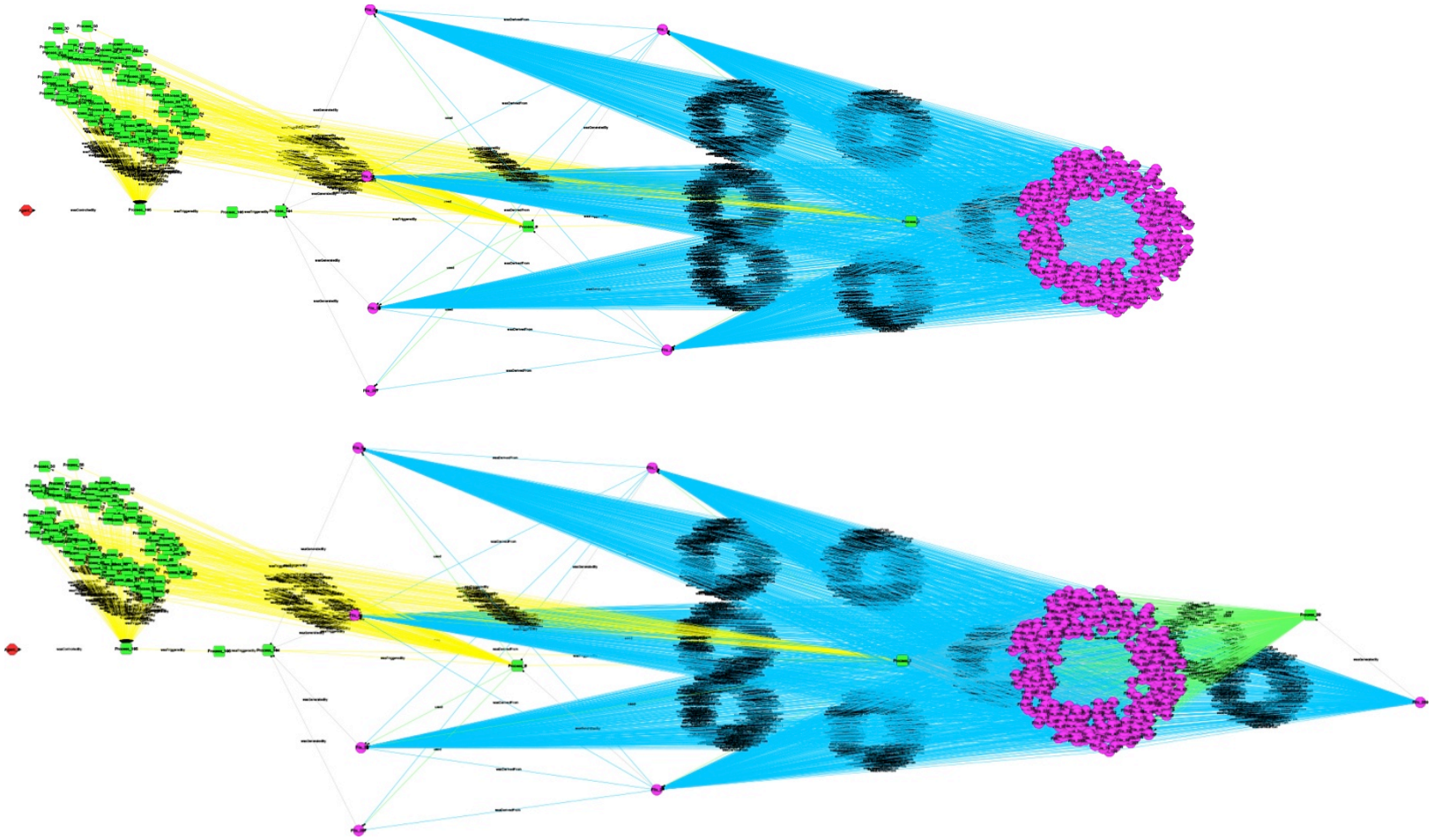


Fig. Layered provenance for BWA. The graphs show the significance and extent of middleware provenance as opposed to total provenance which is comprised of middleware and application layer provenance.

Failure tracing

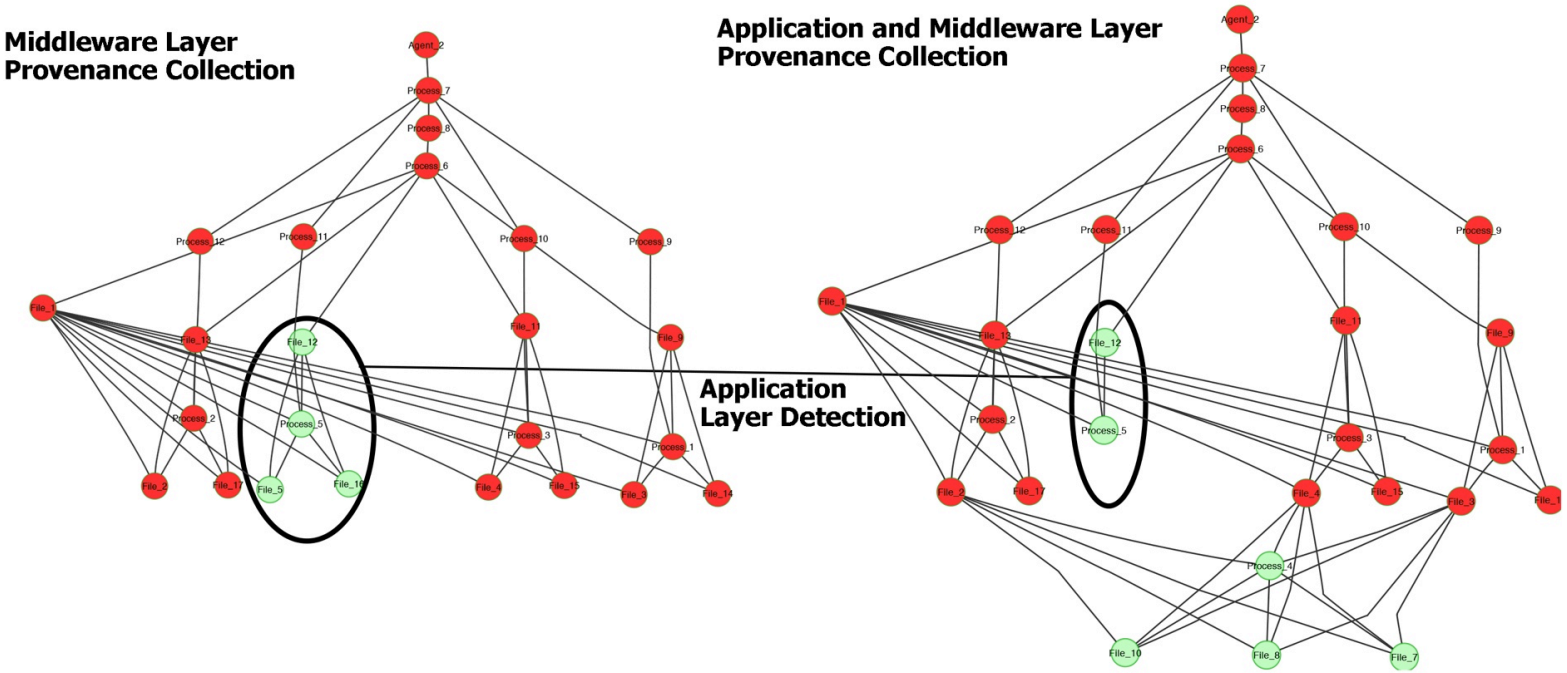


Fig. Application Failure. **Identification of different types of failures using layered provenance capture.**

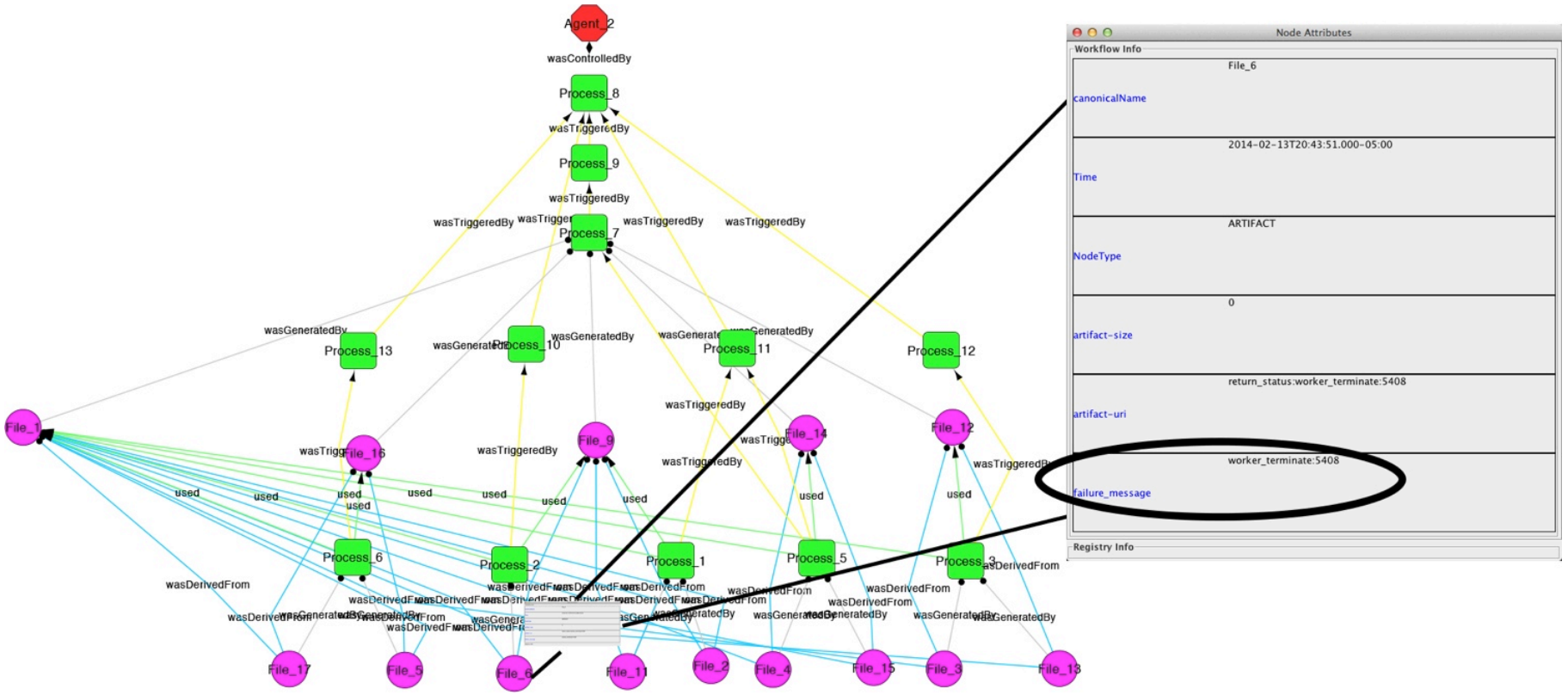


Fig. Middleware Failure. **Identification of different types of failures using layered provenance capture.**

Performance Overheads

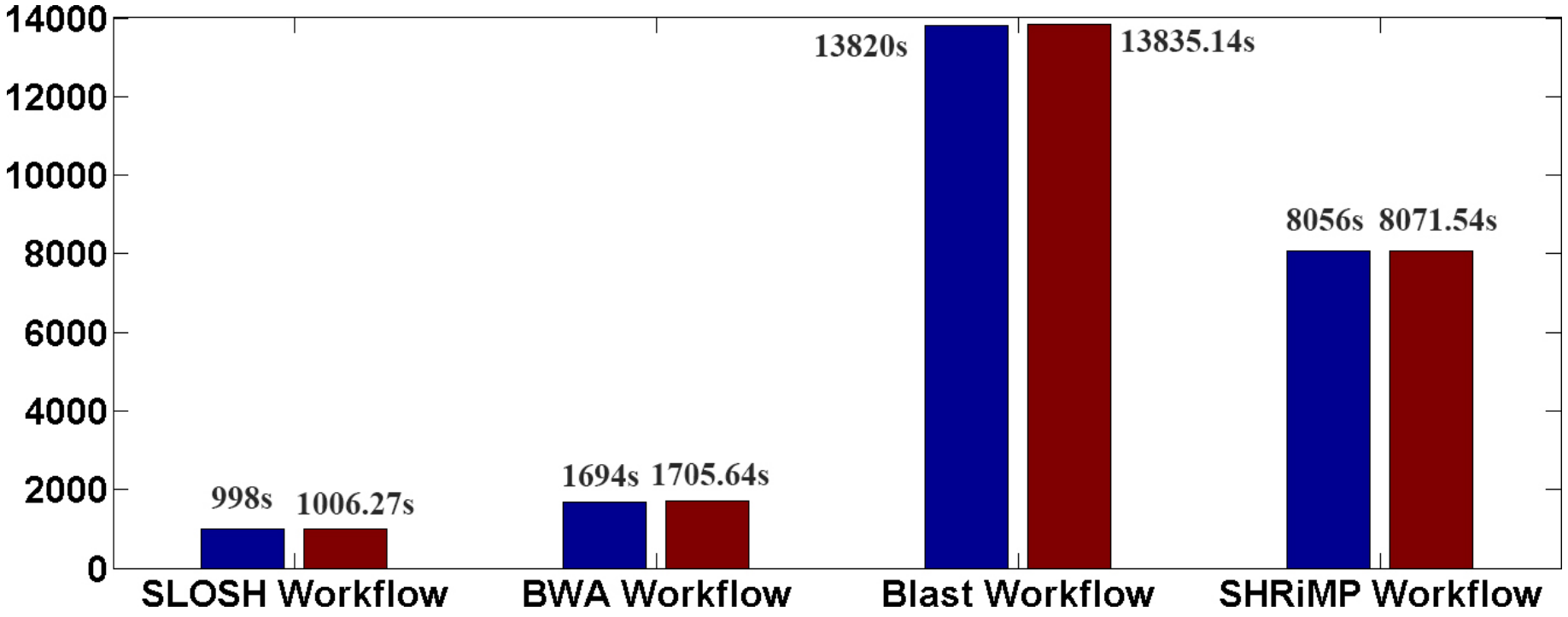


***Fig**. Performance overheads of using provenance-aware Middleware.*
The blue bars *show execution times for different applications using WorkQueue without using the provenance-wrapper.* ***The red bars*** *show the results by making WorkQueue provenance-aware by encapsulating it within our provenance collection framework.*

Completeness Evaluation

	SLOSH		BWA		BLAST		SHRiMP	
	M	M+A	M	M+A	M	M+A	M	M+A
Entities (E)	13	17	105	111	114	117	204	206
Artifacts (D)	11	12	311	313	183	185	10159	10161
Relationships (R)	20	29	522	924	313	642	15178	25399
Annotations (A)	100	127	520	975	635	1025	15837	17120
Completeness-index, C_i	0.81		0.83		0.84		0.91	

Table. *Completeness of middleware provenance ($w_e = 1; w_d = 1; w_a = 0:5; w_r = 0:5$)*

Discussion

- Middleware provenance is able to capture job distribution and execution plans of a workflow with WorkQueue specification that outlines the steps of workflow execution;
- Capturing only from middleware can eliminate certain important elements of provenance; Application layer provenance can be used as enhancement and also to cross-validate the correctness of middleware provenance.

Limitation

- Varied and changing log formats
- Quality of middleware provenance depends on different applications middleware

Conclusion & Future work

- Sources available at: https://github.com/Data-to-Insight-Center/Prov-scaffold

- Extend methodology to other application middleware

- Refine approach by adding correlation inference rules and user controllers

Acknowledgement

- This work is funded by in part by the National Science Foundation under grant OCI-1234983. We thank Douglas Thain of University of Notre Dame, Craig Mattocks of the National Hurricane Center, and Abhirup Chakraborty of Indiana University for help and discussions.

Contact Information

- Quan (Gabriel) Zhou: quzhou@indiana.edu
- Devarshi Ghoshal: dghoshal@indiana.edu
- Beth Plale: plale@indiana.edu

Questions?

Thanks!