PANDA
A System for Provenance and Data

Example: Sales Prediction Workflow

Backward Tracing

Name | Name | Name | Name
-----|-----|-----|-----
Amelia | Isabelle | Jacques | Isabelle

USA | USA
-----|-----

Europe

Catalog Items
Buying Patterns

Item Agg

Union

Split

CustList

Dedup

Sales Predict

Catalog Items Buying Patterns
Example: Sales Prediction Workflow

<table>
<thead>
<tr>
<th>Name</th>
<th>Address</th>
<th>Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amélie</td>
<td>65, quai d'Orsay, Paris</td>
<td>Jacques</td>
<td>39, rue de Bretagne, Paris</td>
</tr>
<tr>
<td>Isabelle</td>
<td>20, rue d’Orsel, Paris</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example: Sales Prediction Workflow

Provenance (very general)

- Where data came from
- How it was derived, manipulated, combined, processed, ...
- How it has evolved over time
Uses for Provenance (very general)

- **Explanation**
  Sources and evolution of data; deeper understanding

- **Verification**
  - Buggy or stale source data? Buggy processing?
  - Auditing

- **Recomputation**
  Propagate changes to affected “downstream” data

Some Application Domains

- **Sales prediction workflows**
- **Scientific-data workflows**
  - Including human-curated data
  - Including evolving versions of data
- **Any analytic pipeline**
- **“Extract-transform-load” (ETL) processes**
- **Information-extraction pipelines**

Third Time’s a Charm

<table>
<thead>
<tr>
<th>Isn’t provenance the same thing as lineage?</th>
<th>Pretty much</th>
</tr>
</thead>
<tbody>
<tr>
<td>Haven’t you worked on it before?</td>
<td>Yes</td>
</tr>
</tbody>
</table>

1. **Data Warehousing project (long ago)**
   - Lineage of relational views in warehouse:
     - formal foundations, system/caching issues
   - Lineage in ETL pipelines: foundations & algorithms

2. **“Trio” project (recently)**
   - Data + Uncertainty + Lineage
   - Lineage primarily in support of uncertainty
Panda’s Ambitions

Previous provenance work tends to be...

- Either data-based or process-based
- Either fine-grained or coarse-grained
- Focused on modeling and capturing provenance
- Geared to specific functions or domains

- Capture both: "data-oriented workflows"
- Cover the spectrum in a unified fashion
- Also support provenance operators and queries
- End with a general-purpose open-source system

Remainder of Talk

- Processing nodes and provenance capture
- Exploiting provenance
  - Built-in operations
  - Ad-hoc queries
  - Other uses
- Current research

Processing Nodes

- Relational
  - Structured, well-understood operations
- Opaque (or semi-opaque)
  - Incomplete knowledge of internals
Provenance Capture — Model

- Ultimate model yet to be determined
  - From bipartite graph to Open Provenance Model
- Goals:
  - Support provenance at spectrum of granularities
  - Mesh data-oriented and process-oriented provenance
  - Composability/transitivity
  - Understandability
- For remainder of talk: think bipartite graph

Provenance Capture — Interface

Processing nodes provide provenance information along with output
Eager or Lazy

Provenance Capture — Interface

- Relational, other well-understood transformations
- Automatic — previous work
- Opaque (or semi-opaque)
  - Provided via provenance interface, or inferred
- Worst Case: No access to fine-grained provenance
### Provenance Operations — Basic

- **Backward tracing**
  - Where did the Cowboy Hat record come from?
- **Forward tracing**
  - Which sales predictions did Amelie contribute to?

### Additional Functionality

- **Forward propagation**
  - Update all affected predictions after customers move from Texas to France

### Additional Functionality

- **Refresh**
  - Get latest prediction for Cowboy Hat sales (only) based on modified buying patterns
  - \(\approx\) Backward tracing + Forward propagation
Provenance Queries

- How many people from each country contributed to the Cowboy Hat prediction?
- Which customer list contributed the most to the top 100 predicted items?

Provenance Queries

- For a specific customer list, which items have higher demand than for the entire customer set?
- Which customers have more duplication — those processed by USA or by Europe?

Provenance Queries

- For a specific customer list, which items have higher demand than for the entire customer set?
- Which customers have more duplication — those processed by USA or by Europe?
- **Query language goals**
  - Declarative ad-hoc queries à la database systems
  - Seamlessly combine provenance and data
  - Amenable to optimization
  - Many interesting optimization issues
Current Research

- **System**
  - Workflow infrastructure
  - Basic provenance capture & operations

- **Algorithms**
  - For refresh problem
    (also foundations & implementation)

- **Theory**
  - Decision problems: eager/lazy, intermediate data
  - Provenance foundations for semi-opaque transformations

---

Current Research

- **System**
  - Workflow infrastructure
  - Basic provenance capture & operations

- **Algorithms**
  - For refresh problem
    (also foundations & implementation)

- **Theory**
  - Decision problems: eager/lazy, intermediate data
  - Provenance foundations for semi-opaque transformations

---

Panda System (version 0.1)

- Arbitrary acyclic workflows
- SQL processing nodes
- Python processing nodes
- Backward-tracing and forward-propagation
Future Work

The large majority of this talk
Stay tuned

Backup slides

Optimization Issues

- Query-driven provenance capture
- Eager vs. lazy provenance capture
  - Space-time & query-update tradeoffs
  - Processing-node dependent
    - Ex: Dedup, ItemAgg
- Retain intermediate data sets?
- Fine-grained vs. coarse-grained
- Approximate provenance
Panda System (version 0.1)

System: Key Features

- Arbitrary acyclic workflows
- SQL processing nodes
  - Automatic provenance capture
- Python processing nodes
  - One-one and one-many transformations only
- Backward-tracing and forward-propagation
### Provenance-Based Refresh

- **Problem**
  - Exploit provenance to efficiently compute the up-to-date value of selected output elements.

- **Challenges**
  - Formally defining provenance & refresh:
    - When is new value a fresh version of old one?
    - Provenance as key
  - When can selective refresh be done correctly and efficiently?
    - Properties of transformations, workflows, provenance
  - Algorithms for wide class of transformations and workflows

See recent paper