The data, they are a-changin’
(ReComp: Your Data Will Not Stay Smart Forever)

Panta Rhei
(Heraclitus, through Plato)

Paolo Missier, Jacek Cala, Eldarina Wijaya
School of Computing Science,
Newcastle University
{firstname.lastname}@ncl.ac.uk

TAPP’16
McLean, VA, USA
June, 2016

(∗) Painting by Johannes Moreelse
Data to Knowledge

- **Lots of Data**
- **Meta-knowledge**
  - Middleware
    - Tools
    - Reference datasets
  - Algorithms
- **Big Analytics Machine**
- “Valuable Knowledge”
The missing element: time

Lots of Data
Big Analytics Machine

Meta-knowledge
Middleware
Algorithms
Tools
Reference datasets

Your Data Will Not Stay Smart Forever

"Valuable Knowledge"
Observe change
- In input data
- In meta-knowledge

Enact
- Reproduce (analytics) processes

Assess and measure
- knowledge decay

Estimate
- Cost and benefits of refresh

Lots of Data → The Big Analytics Machine

Meta-knowledge
- Algorithms
- Tools
- Reference datasets
The ReComp decision support system

- Observe change
- Enact
- Assess and measure
- Estimate
- Impact estimation
- Cost estimates
- Reproducibility assessment
- Re-computation recommendations

Change

Events

ReComp Decision Support System

- Diff(.,.) functions
- utility functions

History of Knowledge Assets and their metadata
1. **Observability** (transparency)
   How much can we observe?
   - Structure
   - Data flow

2. **Change detection**: inputs, outputs, external resources
   Can we quantify the extent of changes? → \( \text{diff()} \) functions

3. **Impact assessment**
   Can we quantify knowledge decay?

4. **Control**: reaction to changes
   How much re-computation control do we have on the system?
   - **Scope**: Which instances?
   - **Frequency**: how often?
   - **Re-run Extent**: how much?

**Provenance**

**ReComp concerns**

- **Diff(...) functions**
- **Utility functions**
- **Change Events**
- **ReComp Decision Support System**

**Impact estimation**
- Cost estimates
- Reproducibility assessment

**Reproducibility**
- Virtualisation
- Smart re-run
This talk: White box ReComp -- initial experiments

<table>
<thead>
<tr>
<th>White box</th>
<th>Black box</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Structure</strong></td>
<td><strong>Dataflow</strong></td>
</tr>
<tr>
<td>(static view)</td>
<td>- eScience Central, Taverna, VisTrails…</td>
</tr>
<tr>
<td></td>
<td>Scripting:</td>
</tr>
<tr>
<td></td>
<td>- R, Matlab, Python…</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Data dependencies</strong></td>
<td><strong>Provenance recording:</strong></td>
</tr>
<tr>
<td>(runtime view)</td>
<td>• Inputs,</td>
</tr>
<tr>
<td></td>
<td>• Reference datasets,</td>
</tr>
<tr>
<td></td>
<td>• Component versions,</td>
</tr>
<tr>
<td></td>
<td>• Outputs</td>
</tr>
<tr>
<td><strong>Cost</strong></td>
<td>• Detailed resource monitoring</td>
</tr>
<tr>
<td></td>
<td>• Cloud → £££</td>
</tr>
<tr>
<td></td>
<td>• Wall clock time</td>
</tr>
<tr>
<td></td>
<td>• Service pricing</td>
</tr>
<tr>
<td></td>
<td>• Setup time (eg model learning)</td>
</tr>
</tbody>
</table>
SVI is a classifier of likely variant deleteriousness:

$$y = \{(v, \text{class}) | v \in \text{varset}, \text{class} \in \{\text{red, amber, green}\}\}$$
OMIM and ClinVar changes

Sources of changes:

- Patient variants → improved sequencing / variant calling
- ClinVar, OMIM evolve rapidly
- New reference data sources

CLINVAR / OMIM relevant changes over time for a patient cohort
(Newcastle Institute of Genetics Medicine)
For each run $i$:

**Observables:**
- Inputs $X = \{x_{i1}, x_{i2}, \ldots\}$
- Outputs $y = \{y_{i1}, y_{i2}, \ldots\}$
- Dependencies $D_{11}, D_{12}, \ldots$

Variable-granularity provenance $\text{prov}(y)$
- Granular $\text{Cost}(y) \rightarrow$ single-block level
- Granular Process structure $P \rightarrow$ workflow graph
White-box provenance

Granular: \( used(P_j, d_{ij}, [prov:role = \text{dep}]), d_{i,j} \in D_i \in D \)

Coarse:

- \( \text{entity}(om, [prov:type = \text{OMIM}', version = \text{v}']) \)
- \( \text{entity}(ph, [prov:type = \text{prov:collection}']) \)
- \( \text{entity}(cv, [prov:type = \text{CV}', version = \text{v}']) \)
- \( \text{entity}(vars, [prov:type = \text{prov:collection}']) \)
- \( \text{used}(PtG, om, [prov:role = \text{dep}']) \)
- \( \text{used}(PtG, ph, [prov:role = \text{input}']) \)
- \( \text{used}(vClass, cv, [prov:role = \text{dep}']) \)
- \( \text{used}(vClass, vars, [prov:role = \text{input}']) \)
A history of runs

History database:

\[ H = \{ h(y, v) = \langle P^v, D^v, x^v, prov(y^v), cost(y^v) \rangle \} \]
• **Scope:** Which instances?
Which patients within the cohort are going to be affected by change in input/reference data?

• **Re-run Extent:** how much?
Where in each process instance is the reference data used?

• **Impact:** why bother?
For each patient in scope, how likely is that any patient's diagnosis will change?

• **Frequency:** how often?
How often are updates available for the resources we depend on?
1. History DB \[ H = \{ h(y, v) = \langle P^v, D^v, x^v, prov(y^v), cost(y^v) \rangle \} \]

2. Measurables changes:

Input diff: \( \text{diff}_{in}(x_i^v, x_i^{v'}) \) \( \rightarrow \) one patient at a time

Output diff: \( \text{diff}_{out}(y_i^v, y_i^{v'}) \) \( \rightarrow \) has the change had any impact?

Dependencies \( \text{diff}_{d}(D_i^v, D_i^{v'}) \) \( \rightarrow \) affects entire cohort \( \rightarrow \) scoping

Example:

\( \text{diff}_{OM}(OM^v, OM^{v'}) = \{ t \in DT | \text{genes}(t, OM^v) \neq \text{genes}(t, OM^{v'}) \} \)

\( \text{diff}_{CV}(CV^v, CV^{v'}) = \)

\( \{ \text{var} \in V | \text{varstatus}(\text{var}, CV^v) \neq \text{varstatus}(\text{var}, CV^{v'}) \} \)

\( \cup CV^{v'} \setminus CV^v \cup CV^v \setminus CV^{v'} \)
The ClinVar diff function – two steps

1. Generic set-theoretic operations:

- **ADDED**
  \{ var \in V | \text{varstatus}(\text{var}, \text{CV}') \not\in \text{varstatus}(\text{var}, \text{CV}) \} \land
  \left( \text{varstatus}(\text{var}, \text{CV}') \in \{\text{‘pathogenic’, ‘benign’}\} \lor \text{varstatus}(\text{var}, \text{CV}) \in \{\text{‘pathogenic’, ‘benign’}\} \right)

- **REMOVED**
  \{ var \in V | \text{varstatus}(\text{var}, \text{CV}) \}

- **RETAINED**
  \{ \text{var} \in V | \text{varstatus}(\text{var}, \text{CV}) \not\in \text{varstatus}(\text{var}, \text{CV}') \}

2. A ClinVar SVI specific filter:

- **REMOVED**
  \{ var \in V | \text{varstatus}(\text{var}, \text{CV}') \} \in \{\text{‘pathogenic’, ‘benign’}\}

- **RETIRED**
  \{ var \in V | \text{varstatus}(\text{var}, \text{CV}) \}

- **ADDED**
  \{ var \in V | \text{varstatus}(\text{var}, \text{CV}') \} \in \{\text{‘pathogenic’, ‘benign’}\}

Up to 99% selectivity
Case 1: Granular provenance

Given observed changes in resources $\text{diff}_d(D^v_i, D'^v_i)$

1. **Scoping:** For each $d_{ij} \in \text{diff}_d(D^v_i, D'^v_i)$

   History instance:
   
   $$h(y, v) = \langle P^v, D^v, x^v, \text{prov}(y^v), \text{cost}(y^v) \rangle \in H$$

   is in the scope $S \subseteq H$ if
   
   $$\text{used}(P_j, d_{ij}, [\text{prov:role} = \text{’dep’}]) \in \text{prov}(y^v)$$

   $P_j$ is added to $P_{\text{scope}}(y)$

2. **Re-run Extent:**

   1. Find a partial order on $P_{\text{scope}}(y)$

   2. Re-run starts from each of the earliest $P_j$ such that their output is available as persistent intermediate result

   see for instance Smart Run Manager [1]

Case 2: Coarse-grained provenance

Scoping: Any instance that depends on any $D_{ij}$ is in scope:

For each

$$d_{ij} \in \text{diff}_d(D^v_i, D^v'_i)$$

$P_{scope} = \{P_j\}$, where:

$\text{used}(P_j, D_i, [\text{prov:role = 'dep'}]), \ d_{ij} \in D_i$

This is trivial for a homogenous run population, but H may contain run history for many different workflows!

Re-run Extent:
The mechanism from the fine-grained case still works
Assessing impact and cost

Approach: small-scale re-comp over the population in scope

1. Sample instances $S' \subseteq S$ from the population in scope $S$
2. Perform partial re-run on each instance $h(y_i, v) \in S'$, generating new outputs $y_i'$
3. Compute $\text{diff}_{\text{out}}(y_i^v, y_i'^v)$
4. Assess impact (user-defined) and $\text{cost}(y')$
5. Estimate cost difference $\text{diff}(\text{cost}(y), \text{cost}(y'))$
ReComp is a Decision Support System

Impact, cost assessment $\rightarrow$ ReComp user dashboard
Implementation in progress
Small scale experiments on scoping / partial re-run
- Test cohort of about 50 (real) patients
- Short workflows runs (about 15 mins), observable cost savings
- (preliminary results)

Main challenge: deliver a generic and reusable DSS

From eScience Central \(\rightarrow\) To generic dataflow, scripting (Python)

From
eSc prov traces \(\rightarrow\) PROV-compliant but idiosyncratic patterns
Python \(\rightarrow\) noWorkflow traces

To: Canonical PROV patterns + queries + H DB implementation

ReComp: [http://recomp.org.uk/](http://recomp.org.uk/)


