Report on the First International Workshop on Incremental Re-computation: Provenance and Beyond

Paolo Missier
School of Computing,
Newcastle University, UK
paolo.missier@ncl.ac.uk

Tanu Malik
School of Computing
DePaul University
tmalik1@depaul.edu

Jacek Cala
School of Computing,
Newcastle University, UK
jacek.cala@ncl.ac.uk

1. INTRODUCTION

In the last decade, advances in computing have deeply transformed data processing. Increasingly systems aim to process massive amounts of data efficiently, often with fast response times that are typically characterised by the 4V's, i.e., Volume, Variety, Velocity, and Veracity. While fast data processing is desirable, it is also often the case that the outcomes of computationally expensive processes become obsolete over time, due to changes in inputs, reference datasets, tools, libraries, and deployment environment. Given massive data processing, such changes must be carefully accounted for, and their impact on original computation assessed, to determine how much re-computation is needed in response to changes.

A core challenge is how to optimise re-computation in the presence of changes, given an existing process execution baseline. Specific research questions include (1) how, and under what assumptions, can re-computation be optimised using incremental and/or partial processing techniques given the baseline, and (2) how do we determine the impact of a set of changes on the outcomes, in order to decide when changes should trigger re-computations.

In this article we report on the proceedings of the *First International Workshop on Incremental Re-computation: Provenance and Beyond* (IRPb), which was organised to explore the breadth and depth of the re-computation problem, with specific emphasis on the role of provenance in this area. Within this scope, the workshop provided a forum for experts to constructively explore theoretical, systems-oriented, and provenance-related challenges in developing and using incremental re-computation based systems.

IRPb was held in conjunction with ProvenanceWeek 2018, a bi-annual week-long event that includes the 7^{th} edition of the International Provenance and Annotation Workshop (IPAW), and the 10^{th} Usenix Workshop on the Theory and Practice Of Provenance (TAPP). The format chosen for the workshop was designed to encourage discussion without requiring a paper contribution, other than an abstract. Held over two half-days, IRPb consisted of a collection of 12 short talks (15-20') plus ample time for discussion, given by recognised experts in the areas within the scope, and 2 longer keynote talks. Abstracts and presentations are available at `https://tinyurl.com/y7c8vttn`.

2. WORKSHOP TOPICS

With each of the 14 contributors presenting their own perspective on the topic, we have used the following categories to characterise the contributions, using tags to annotate the individual talks.

Re-computation, i.e., the repeating of a process execution, all or in parts, under slightly different inputs or configuration each time, and making use of one or more prior execution baselines as a basis for optimization. We use tags **#howto-recomp** and **#using-recomp** to distinguish research that describes techniques that advance the state of the art on performing re-computation, from research that makes use of such techniques, respectively.

Incremental computation. This is naturally viewed as one of the ways re-computation can be optimised, however it is arguably more general, as it does not require a prior baseline (first time executions may be incremental). As before, we use tags **#howto-incr-comp** and **#using-incr-comp**.

Approximate computation, a well-established field is identified using tag **#howto-approx-comp**.

Provenance, including all phases of its lifecycle, namely capture, storage, query, and analysis. Again, we make a distinction between **#using-provenance** and **#for-provenance**.

Contributions also covered a diversity of applications areas, ranging from the Life Sciences (genomics and metagenomics), machine learning, data journalism, transportation science, and large-scale

simulations, as well as research areas ranging from databases and data integration, programming languages, reproducibility of e-science processes (scientific workflows, including workflow steering), process mining, and naturally, core provenance research.

3. CONTRIBUTIONS

3.1 *Keynote:* Language-based issues in incremental computation

James Cheney's talk presented the different forms on incremental computation and their use in programming languages. He outlined work (mostly by others) in three sub-areas (i) on incremental, (ii) self-adjusting and (iii) bidirectional computation, clarifying the meaning and subtle formalism difference within each area. James presented a simplified example of a delta data structure used to compute the square of the sum of two numbers, and described how such data structure is maintained using static differentation and more involved incremental lambda calculus as described in the seminal paper of Cai et. al. [2]. Self-adjusting computation implies recomputing efficiently as the input is changed, using caching to avoid recomputation of sub-expressions whose results have not changed. James covered work from Acar et.al [4] and described the primary idea of self-adjusting traces, which first use execution to generate annotations (the trace), and then use the annotated program trace to make subsequent runs probably faster compared to running "from scratch". He highlighted the opportunity of including provenance-like information in these traces to improve incremental computation. The limitation of the approach is that generating annotations often requires slight program modification, but also mentioned significant progress on how to reduce the annotations. Finally, he covered bidirectional computation, which means updating the input to a computation to be consistent with a proposed new output: a generalization of the view update problem. In this he presented a recent contribution to incremental view update problem in relational databases, which is of interest for computing "missing answer" or hypothetical explanations for database settings.

3.2 *Keynote:* Modern Dataflow

Frank McSherry's keynote talk summarized his recent work on Differential Dataflow [11], a data-parallel programming and execution model for scalable and incremental computation, which is based on the 'Timely Dataflow' framework [12] and a data serialization library called 'Abomonation'.

Differential Dataflow is a collection-oriented programming model in which users program their algorithms with a set of functional operators, while the system manages incremental changes to the input data. The set of operators includes well-known data-parallel functions like: map, reduce and join but also iterate, which allows *incremental and iterative* algorithms to be implemented. The key element of the model, which makes it distinct from other approaches to incremental computation, is that in Differential Dataflow the state of computation and its updates are associated with a multi-dimensional logical timestamp. Such association allows the system to maintain a partially ordered set of versions (data updates) rather than only the most up-to-date coalesced state of computation.

Frank compared the efficiency of his platform and showed at least an order of magnitude speed-up for graph-based problems like PageRank and connected components. He also illustrated computation on directed acyclic graphs in which 40% of the computation does not change the output due to change in inputs, whereas 60% changes it only moderately.

The talk also briefly introduced two other elements of the Modern Dataflow platform: the Timely Dataflow framework, to scale the same program up from a single thread to distributed execution on a cluster of machines, and the Abomonation library for fast data serialization, in the Rust language.

3.3 Short Presentations

Answering Why-Not queries Against Scientific Workflow Provenance Khalid Belhajjame (University Paris-Dauphine, France) focused on a variation of the well-known problem of answering *Why-not* database queries, that is, to explain why a certain tuple is not returned as part of a query answer. The problem finds a similar formulation but requires a new approach when the question is posed on the result of a workflow execution. The proposed approach is shown to require the re-computation of parts of the workflow. **#using-recomp**, **#for-provenance**.

The Marriage of Incremental and Approximate Computing Pramod Bhatotia (University of Edinburgh) described differences between incremental and approximate computation. In essence, both paradigms rely on computing over a subset of data items instead of computing over the entire dataset, but they differ in their means for skipping parts of the computation. Pramod suggests that the two approaches are complementary and can be com-

bined, namely by using a stratified sampling algorithm that biases the sample selection to the memoized data items from previous runs. The resulting implementation, based on Apache Spark Streaming, is part of a data analytics system called *IncApprox* (Incremental + Approximate Computing) [9]. **#howto-incr-comp**, **#howto-approx-comp**.

Supporting Incremental Re-Computation with Whole System Provenance: Issues and Approaches Ashish Gehani (SRI International, USA) focused on *whole system provenance*, that is, provenance collected from observations of system-level events during process execution, as a way to make process re-computation efficient, i.e., by reducing the fraction of computation that needs to be performed again. The talk addressed practical issues that arise in this context, and outlined approaches to address them. The challenges include ephemeral intermediate artifacts, conflated causality, dynamic runtime environments, and external dependencies. **#howto-recomp**, **#using-provenance**

TensorCell - approximating outcomes of computer simulations using machine learning algorithms. Pawel Gora (University of Warsaw, Poland) presented an approach for approximating the result of traffic simulations using neural networks. Simulating the collective behaviour of traffic lights in a large city, such as Warsaw, is computationally complex. Exploring "what-if" scenarios by repeating the simulation becomes prohibitive. The authors tested the hypothesis that reasonable approximations of the expected waiting time at each of the lights can be obtained by training a neural network on an exemplar set of light configurations (on 15 significant crossroads). The results are encouraging, with average prediction error varying between 1.18% and 6.8%, with sub-second processing time on the network. **#howto-approx-comp**.

Progressive Provenance Capture Through Re-computation Paul Groth (Elsevier Labs, NL) described his research on using record-replay technology within virtual machines to incrementally add additional provenance instrumentation by replaying computations after the fact. **#using-Recomp**, **#for-provenance**

Incremental Recomputation in Data Integration Melanie Herschel's (University of Stuttgart, Germany) contribution focused on complex data integration processing pipelines, which are often developed and maintained incrementally, and require multiple iterations to reach satisfactory results. The talk discussed the potential benefits of how-provenance and what-if analysis in supporting incremental re-computation. **#using-incr-comp**, **#using-provenance**

Incremental Recomputation: Those who cannot remember the past are condemned to recompute it Bertram Ludascher (University of Illinois at Urbana Champaign, USA) explored the connection between re-computation, Models of Computation (MoC), and *models of provenance* (MoP). He made the point that "computing with deltas" has been common for a variety of "Models of Computation", from Datalog (Delta Computations and semi-naive evaluation, Statelog (Stateful Datalog)) to incremental view maintenance in databases, to workflow programming. In particular, a connection has been made over the years between MoCs associated with workflows, for example when using Kepler [10], and corresponding MoPs [1], and can also serve as a foundation for implementing provenance-based fault tolerance mechanisms [8], i.e., using checkpoints and partial re-run. **#howto-incr-comp**, **#howto-recomp**

Incremental Recomputation in Containers Tanu Malik (DePaul University, USA) described the need for incremental computation in reproducible computation. She showcased Sciunit (`https://sciunit.run`) reproducible containers, which capture necessary and sufficient binaries and data so as to repeat the computation in a new environment, but must be entirely re-evaluated every time a change to input parameter or dataset is made [13]. She highlighted the need of incremental re-computation techniques on versioned provenance graphs. **#howto-recomp**, **#using-recomp**.

Collecting Provenance of Steering Actions Mattoso and Sousa (UFRJ, Brasil) presented *workflow steering* [14], a form of human-in-the-loop scientific workflows where experts are given the chance to repeatedly tune some of the process' parameters at runtime, with the aim to significantly improve execution performance or improve result quality. The problem addressed in the talk is how to track and record users' steering actions, i.e., using a provenance-based framework. **#howto-incr-comp**, **#using-provenance**.

Provenance and recomputing in the realm of large scale environmental sequence analysis Folker Meyer (Argonne National Labs, USA) talked about the needs for re-computing in the realm of large scale environmental sequence (metagenomics) analysis, where frequent changes in the underlying knowledge databases render in-silico analy-

sis results both uncertain and unstable. Taking the perspective of a large-scale analysis provider (*MG-RAST*, `https://mg-rast.org`) which caters to tens of thousands of scientists from many domains, the talk highlighted the need for effective re-computation tools. **#using-recomp**, **#using-provenance**.

The ReComp project: an overview Paolo Missier (Newcastle University, UK) started from the observation that the outcomes of computationally intensive processes (data processing pipelines, simulations) are often time-sensitive, as they depend on algorithms, tools, and reference databases that evolve over time. A re-computation problem naturally occurs when some of the changes in these elements invalidate some of the outcomes. The problem is to estimate which of the past outcomes are affected by a change, and to what extent. The talk provided an overview of *ReComp* (`http://recomp.org.uk`), a generic framework designed to determine the minimal process fragment that requires re-computation [3]. **#howto-recomp**, **#using-recomp**.

Handling late data in process mining algorithms Tomasz Pawlowski and Jacek Sroka (University of Warsaw, Poland) situate their research at the intersection of process mining, a very mature research area, and stream data processing. They observe that, while a number of process mining techniques exist analyse and visualise repetitive processes, those mostly operate offline, on static event logs. They focus instead on online analysis of streams of logs, where problems occur when some events are logged out-of-order. For instance, out-of-order events happen when a streaming process is offline. The authors investigate how in this setting incremental re-computation of existing process mining algorithms occurs—in particular to handle out-of-order data without repeating the whole data mining computation from scratch. **#howto-incr-comp**.

Self-Explaining Computation with Explicit Change Perera (University of Edinburgh, UK) proposed a notion of *self-explaining computation with explicit change*. He used Jupyter notebooks as an example of *data-driven storytelling* that can offer some form of *explorable explanations*. But Prera mentioned there are limitations with respect to transparency of explainations. The goal of self-explaining computation with explicit change is to increase transparency by making it explicit how parts of a computation relate to other parts, and how changes cause other changes. The latter point connects this research with incremental (re-)computation, with the idea to leveraging ideas from partial and incremental computation as well as self-explaining computation [4] [6, 7] to user interfaces, namely by enabling components which use provenance [5] to support slicing and delta-visualisation, making explanations accessible directly from data views. **#using-incr-comp**

4. REFERENCES

[1] Anand, M. K., Bowers, S., McPhillips, T. M., and Ludäscher, B. Exploring Scientific Workflow Provenance Using Hybrid Queries over Nested Data and Lineage Graphs. In *SSDBM* (2009), pp. 237–254.

[2] Cai, Y., Giarrusso, P. G., Rendel, T., and Ostermann, K. A theory of changes for higher-order languages: Incrementalizing λ-calculi by static differentiation. In *ACM SIGPLAN Notices* (2014), vol. 49, ACM, pp. 145–155.

[3] Cała, J., and Missier, P. Selective and Recurring Re-computation of Big Data Analytics Tasks: Insights from a Genomics Case Study. *Big Data Research in press* (aug 2018).

[4] Cheney, J., Acar, U. A., and Perera, R. *Toward a Theory of Self-explaining Computation*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 193–216.

[5] Cheney, J., Chiticariu, L., and Tan, W.-C. Provenance in databases: Why, how, and where. *Foundations and Trends in Databases 1*, 4 (2009), 379–474.

[6] Griffin, T., Libkin, L., and Trickey, H. An improved algorithm for the incremental recomputation of active relational expressions. *IEEE Transactions on Knowledge and Data Engineering* (1997).

[7] Horn, R., Perera, R., and Cheney, J. Incremental relational lenses. *Proc. ACM Program. Lang. 2*, ICFP (July 2018), 74:1–74:30.

[8] Köhler, S., Riddle, S., Zinn, D., McPhillips, T., and Ludäscher, B. Improving workflow fault tolerance through provenance-based recovery. In *SSDBM* (2011), pp. 207–224.

[9] Krishnan, D. R., Quoc, D. L., Bhatotia, P., Fetzer, C., and Rodrigues, R. Incapprox: A data analytics system for incremental approximate computing. In *25th International Conference on World Wide Web* (2016), pp. 1133–1144.

[10] Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E. A., Tao, J., and Zhao, Y. Scientific Workflow Management and the Kepler System. *Concurrency and Computation: Practice and Experience 18*, 10 (2005), 1039–1065.

[11] McSherry, F., Murray, D. G., Isaacs, R., and Isard, M. Differential dataflow. In *CIDR* (2013).

[12] Murray, D. G., McSherry, F., Isaacs, R., Isard, M., Barham, P., and Abadi, M. Naiad: a timely dataflow system. In *Proceedings of the Twenty-Fourth ACM Symposium on Operating Systems Principles* (2013), ACM, pp. 439–455.

[13] Pham, Q., Malik, T., and Foster, I. Using Provenance for Repeatability. In *TaPP* (2013).

[14] Souza, R., and Mattoso, M. Provenance of dynamic adaptations in user-steered dataflows. In *IPAW* (2018), pp. 16–29.