# Provenance: A Future History

James Cheney
University of Edinburgh
jcheney@inf.ed.ac.uk

Stephen Chong
Harvard University
chong@seas.harvard.edu

Nate Foster
University of Pennsylvania
jnfoster@cis.upenn.edu

Margo Seltzer
Harvard University
margo@eecs.harvard.edu

Stijn Vansummeren *
Hasselt University/Transnational Univertsity of Limburg, Belgium
stijn.vansummeren@uhasselt.be

## Abstract

Science, industry, and society are being revolutionized by radical new capabilities for information sharing, distributed computation, and collaboration offered by the World Wide Web. This revolution promises dramatic benefits but also poses serious risks due to the fluid nature of digital information. One important cross-cutting issue is managing and recording *provenance*, or metadata about the origin, context, or history of data. We posit that provenance will play a central role in emerging advanced digital infrastructures. In this paper, we outline the current state of provenance research and practice, identify hard open research problems involving provenance semantics, formal modeling, and security, and articulate a vision for the future of provenance.

***Categories and Subject Descriptors*** D.3.1 [*Programming Languages*]: Formal Definitions and Theory

***General Terms*** Languages, Security, Reliability

***Keywords*** provenance, integrity, semantics

## A future history of provenance

*Transcribed keynote remarks from the 2019 Federated Provenance Conference, by [inaudible]*

We won! Provenance is everywhere: it's part of every storage system and every application; it's secure; it's queryable; it's indexed by standard search engines and can be queried anywhere.

Provenance is pervasive, invisible, and handled automatically by default. Ten years ago people regularly used data without knowing where it was hosted or originated. Today such a thing is unthinkable. Documents whose origins are unknown are highly suspect and quickly attract the attention of a small but enthusiastic anti-plagiarism community.

Both public and confidential provenance are incorporated into the algorithms employed by major search engines, who use the provenance of a page to weigh its reliability, and direct searches toward more reputable results. Individuals are able to query where their data is being used elsewhere on the Web, and can derive economic benefit and acclaim by providing useful data. The Semantic Web only became truly compelling once people had an incentive for contributing high-quality metadata. Provenance tracking provided this motivation by ensuring recognition for contributors. Some curators even make a living managing Semantic Web metadata.

The old vertically-integrated media industries (particularly music and print media) loved provenance—for a while. They thought it would revive their dream of complete control over their intellectual property through digital rights management. Indeed, provenance is now used to ensure that authors and creators are recognized and rewarded for their work, whether they are independent or affiliated with a major media label or news agency. The key difference between the success of provenance tracking and the failure of earlier attempts at DRM is that provenance is based on voluntary participation in the online community, not on centralized control imposed by legislation or lawsuits. Creators are free to decide whether to retain or expose provenance, and recipients of such information are free to decide whether to believe or trust it.

But how did we get here?

* Stijn Vansummeren is a Postdoctoral Fellow of the Research Foundation—Flanders (Belgium).



*OOPSLA 2009,* October 25–29, 2009, Orlando, Florida, USA.

By 2009, it was widely recognized that provenance was important, but the effects of Moore's Wall were only beginning to be felt, and 20th-century assumptions about storage and computational overhead still held sway. The cost of storage was still relatively high—around $20 for a terabyte instead of pennies. Likewise, typical personal computers contained only a few cores rather than the hundreds or thousands now common today, and reliable, usable concurrent programming languages had not yet become mainstream. Now, of course, the extra storage and computational overhead of instrumenting programs to record and maintain detailed provenance metadata is effectively zero. Indeed, provenance pays for itself in terms of decreased liability for failures—most insurers today won't even cover businesses that use hardware or software that is not provenance aware.

A thornier issue was how to manage and search the masses of detailed provenance information that began to be generated. Early work viewed provenance as a simple directed acyclic graph. While rich languages for querying graph-like data had already been researched, they had not yet been integrated into scalable, robust, and industrial-strength systems. However, the development of DBMSs supporting fast graph queries was only a first step towards the rich provenance query languages we have today. Provenance queries essentially query the behavior of programs, and it was a significant challenge to formulate nontrivial provenance queries manually over the raw, low-level representation as a DAG. Instead, the first half of this decade saw the development of advanced provenance query languages that incorporated ideas from reflective and aspect-oriented programming. These languages make it easy for programmers to query provenance using the syntax they already use to write programs. A nice side-benefit of this was that major programming languages and implementations finally took database–programming language integration seriously, solving the "impedance mismatch" problem once and for all.

Some cynics admitted that provenance might be important for critical applications such as banking or medical records, but doubted that it would carry over to general-purpose applications: "as soon as you move an object from one system to another," they said, "you're going to lose its provenance—we can't even do extended attributes in a portable way!" But researchers and developers from a host of different areas banded together and formed a "provenance community". Much like in the early days of the Internet and World Wide Web, their grass-roots efforts led to the rapid development and implementation of de facto standards for representing and transporting provenance metadata. These were eventually codified by the W3C and other organizations.

The provenance tide began to turn when the financial industry imploded in 2008. In the wake of this disaster, which raised the stakes for regulatory oversight, the US Congress passed the 2010 Oversight Act which mandated that every major financial transaction have a verifiable record. This had an energizing effect on the provenance community similar to that of the Department of Defense's Orange Book on the security community in the 1970s. Other nations followed suit, legislating stronger standards for transparency for both financial and social policy data.

Security and privacy also had to be completely rethought with the adoption of pervasive provenance technology. Nowadays, users can limit access to both sensitive data and its provenance, at the cost that others may be less willing to believe or trust their data. The tension between privacy and provenance was graphically illustrated by the protests in Iran following the contested election of June 2009. Sites such as Twitter, Facebook and YouTube played an essential role early on, making it possible for protesters to connect with each other and journalists to report to the outside world without government interference, but there was also a great deal of unverifiable "noise" and deliberate misinformation. Moreover, this information (along with more traditional forms of surveillance) was later used effectively to single out prominent Facebook or Twitter users for persecution in the brutal repression that followed the protests.

In the rest of this talk, I'll ask you to think back to the summer of 2009. It was a pivotal time for this field. Many of these ideas were in their infancy, and almost no one had any idea how important provenance would become over the next 10 years. In fact, many computer scientists had either not heard of provenance, or thought it was either trivial or outright impossible. Some early papers on provenance were vague about either what they wanted to accomplish, or what they were actually proposing as solutions. Many papers proposed solutions to related problems, but did not provide enough detail about either solution or problem to make a real comparison possible. It was quite difficult for newcomers to the field to understand what was really essential. General foundational definitions, clear problem statements, and theories of provenance only started to coalesce by the end of the last decade. These ideas played a central role in all of the above developments, because a cohesive provenance community would have been impossible before everyone was able to agree on basic definitions. Perhaps the most important insight was...

*[The rest of the recording is inaudible.]*

## 1. Introduction

In an ideal world, software systems would be engineered to the highest standards. Programs would be expressed in intuitive high-level languages and their behavior would be checked against clean formal specifications. Data would be classified according to precise schemas and curated with accurate metadata. But we do not live in that ideal world. Few real-world systems meet these lofty standards. In practice, programs are often built on top of legacy code and dirty data containing errors, omissions and outright lies.

Of course, these problems are not new—the difficulty of building reliable software has long been known—but they have recently become even more urgent as technologies such as "the Semantic Web", "Grid middleware", "cloud computing", "ubiquitous computing", "eScience", "cyberinfrastructure", "e-Infrastructure", etc. [21] start to take hold. These emerging technologies build on the fabric provided by the Internet and the World Wide Web to enable fundamentally new kinds of interaction interact, collaboration and computation. For example, eScience and cyberinfrastructure systems are intended to give scientists easy access to high-performance computing, help them create high-quality *curated databases*, and facilitate collaboration using social-networking tools and *virtual research environments*.

But while these technologies offer dramatic advantages, they also exacerbate the hazards posed by buggy programs and dirty data. Digital information is easy to copy, change, and misinterpret. Current software systems do not provide the levels of repeatability, reliability, accountability and integrity achieved by the paper-based technology—books, academic journals and laboratory notebooks—that they are predicted to replace [23].

Our thesis is that successfully realizing the visions(s) of these revolutionary computing technologies will require building *provenance technology* into all computer systems of any importance. By *provenance*, we mean information about the origin, context, or history of the data. By *provenance technology*, we mean the hardware and software components that are needed to record and maintain robust provenance as an end-to-end resource in a system. Provenance will help recover the repeatability, integrity and authenticity properties of pre-digital information, and will also make it easier to detect and prevent failures, analyze errors, and discourage malfeasance by increasing transparency and accountability.

Without pervasive adoption of provenance technology, critical computer systems and networks have serious vulnerabilities, which we call *provenance failures*. Today, such failures are frequent, costly and embarrassing. Here are just a few illustrative examples:

- In December 2006, a biochemist retracted five papers after discovering bugs in a data analysis program that had been used to generate several years of research results [26]. The use of sophisticated Grid, database and other systems provides many new opportunities for similar errors to creep into the scientific record.
- In September 2008, an undated article concerning United Airlines' near bankruptcy [12] appeared on the list of top stories on Google News. The story was no longer relevant—it had originally been published six years earlier—but investors panicked, and the share price of the company fell by 75% before trading was halted.
- In February 2009, the US Congress passed into law a $800 billion stimulus package. Changes were made by hand on paper copies of the bill only hours before it was enacted, including changes to the amounts allocated to specific programs. For example, a $3.4 billion allocation to energy conservation programs was changed to $3.1 billion in the final draft (see Figure 1).

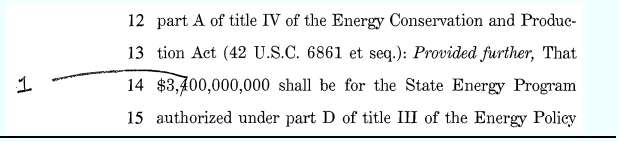
12 part A of title IV of the Energy Conservation and Produc-
13 tion Act (42 U.S.C. 6861 et seq.): *Provided further,* That
14 $3,400,000,000 shall be for the State Energy Program
15 authorized under part D of title III of the Energy Policy

**Figure 1.** The pen is mightier than the sword.

- Prior to the 2003 Iraq invasion, the UK government published Microsoft Word documents containing supposedly objective intelligence evaluations. However, the documents' revision history disclosed that some of the authors were political operatives [33].
- The current financial crisis, which began in late summer 2008, is partly attributable to increased opacity of financial markets due to deregulation and large-scale computerization over the last twenty years. Lewis and Cohan write in the New York Times:

  > Ever since traders started disappearing from the floor of the New York Stock Exchange in the last decade of the 20th century, there has been less and less transparency about the price and volume of trades. [22]

  They go on to call for the US government to mandate that regulators provide impartial, transparent and trusted information about transactions.

Generally speaking, a provenance failure is a negative outcome caused by a failure to record, manage, interpret, or control access to information and its provenance. These failures are preventable, but currently prevention requires a great deal of vigilance (and effort) from users and requires cooperation across multiple systems and domains of control. Provenance failures will either continue to cause major losses if critical systems are developed on top of advanced, but unreliable infrastructure, or will, ipso facto, ensure that such infrastructure is not used for anything important.

Provenance failures overlap with other well-known kinds of failures in computer systems, such as hardware failures, software bugs that lead to incorrect (or misleading) results, and security vulnerabilities that allow misuse (or prevent legitimate use) of computer systems. However, provenance failures also differ from each of these in important ways. In particular, provenance failures involve *users' expectations* or *(mis)interpretations* of data in a system relative to real-world artifacts or situations. Provenance failures thus do not appear to map naturally to the existing goals of correctness or efficiency. Moreover, although provenance is similar to security in that it is a hard-to-define, end-to-end property of a system, existing security models do not yet seem to help us

to understand provenance either. Therefore, we believe that accurately modeling and effectively combating provenance failures requires new ideas.

Unfortunately, the challenge of provenance is often underappreciated. Provenance seems trivial, in theory: as a thought experiment, one can just imagine recording everything that might be useful later on. Of course, this is not feasible in practice; however, it is also problematic in theory, for it begs the question of how to define "everything"—when do we stop? Likewise, given an idealized, but impractical definition of "everything", how can we compare practical techniques that fall short of this ideal? These questions seldom receive the attention they deserve. Instead systems (or users) that record provenance typically do so in an ad hoc way based on perceived needs or as a reaction to known problems. In the absence of agreement about even basic definitions, goals, and success criteria for provenance techniques, it is hard to see how we can do better.

High-performance computing, formal verification, and security are widely appreciated to be challenging. They each have a received a great deal of attention within cohesive research communities, have well-developed theoretical foundations, and have made significant practical impact. The study of provenance currently enjoys none of these features, although it is increasingly recognized as a hard problem: in the US it has been included on the InfoSec Council's Hard Problems List [3], and in the UK it has been identified as a Grand Challenge by the British Computing Society [2]. Nevertheless, there is little broad understanding of what provenance is, what problems it solves, or what the key open problems or challenges are.

This paper is a call to arms for provenance-related research, outlining directions that we believe are understudied and drawing attention to some of the hard problems involving provenance, in the hope the accomplishments anticipated in our opening remarks may become reality. This paper is *not* an attempt at a complete or fair survey of the field; we refer interested readers to other recent surveys [7, 29] and resources [1].

We begin in Section 2 by giving a high-level overview of provenance and its potential for digital information. In Section 3 we discuss some of the problems with the current treatment of provenance, while in Section 4 we outline our views of the key challenges and open problems that must be addressed in order to develop robust provenance technology. We conclude in Section 5.

## 2. What is provenance and what is it for?

What is provenance? That depends on the context in which the question is posed, and on the goals provenance is expected to achieve. Many different user communities are calling for some metadata called "provenance", but each of these communities actually has different underlying needs and applications in mind [1]. Moreover, different communities think of their objects and data at different semantic levels. So just like any programming or data management problem, we should ask what provenance is, what problems it solves, and how we shall recognize success.

The concept of "provenance" originates from the art and archiving worlds, where it refers to information about the creation, chain of custody, modifications or influences pertaining to an artifact.

In the art world, such documentation is important for being able to judge *authenticity*, that is, whether a work is what its owners/sellers claim it is (rather than a forgery). Thus, it is important that the record be as detailed as possible, and that it be possible to check the record against other, independent records as well as against the artifact itself.

In archiving, provenance is related to *integrity*, such as the principle that archivists should respect the order or organization of a collection of documents that are being archived: this principle is called *respect des fonds*. This is in contrast to archiving practices popular in earlier periods (e.g., 19th century) when archivists often devised their own scheme for reorganizing collections. Doing so discards information about the relationships among objects that might not be apparent to the archivist and might be difficult for others to recover after reorganization.

A more concrete answer to the question "what is provenance for *digital* artifacts" is to look at features or applications of current computer systems that appear related to history tracking, logging, integrity, authenticity, or error recovery. Most computer systems, and many users, currently track such information to aid in error correction, debugging, auditing, or just as simple memory aids:

- Operating systems log important system events, to aid system administration and intrusion detection.
- File systems record basic metadata such as file creation, modification, ownership and permissions.
- Version control systems record metadata about when changes have been made and by whom.
- Compilers use source line number tagging to aid in pointing to the sources of compile-time or run-time errors.
- Scientific database curators maintain detailed high-level records of who has inserted, modified or deleted data, and (sometimes) where it has been copied from.
- Web browsers retain history information about which sites have been visited and when.
- Users of Twitter, Friendfeed and other social networking systems employ informal conventions concerning acknowledging sources of quoted data or links, e.g. the abbreviations "RT" (ReTweet) or "HT" (Hat Tip).

Although each of the above represents an example of provenance, we do not believe any of the above mechanisms provides a self-contained definition of provenance, even within its context. In each case, we can imagine taking the exist-

ing system and modifying it to record more information that could be considered provenance, though again this would quickly become impractical. Thus, each of the above mechanisms represents a practical summary of the true provenance of the system, a concept which we posit exists in each case but won't try to define.

Now what is provenance for? We have given two typical motivations above, namely for assessing authenticity and integrity. The above examples also suggest distinguishing two complementary motivations for recording provenance: failure recovery and success validation.

Any computer system can fail in a number of ways: there can be a hardware fault, software bug, malicious user, or simple human error. When a failure occurs, we need to know what happened, how the failure occurred, who was involved, who was to blame, and how safeguard against similar failures in the future. Conversely, if a system is being used to make decisions upon which significant resources or lives depend, then it is important for the process leading to a particular decision to be transparent, comprehensible, and persistent. Such decisions need to be justified by explicitly showing how the results were derived, what assumptions or approximations were used, who was involved, who deserves credit, and how to reproduce the results in different circumstances.

Scientists are required to imagine (and defend against) all the possible ways their experiments might fail or might mislead us into drawing flawed conclusions. Accordingly, laboratory scientists have developed careful record-keeping practices that make it easier for a scientist to satisfy herself (and to convince others) that her work is valid, repeatable, and accurate. These records anticipate scientists' need for both success validation and failure recovery in the course of responsible conduct of research [24].

Failure recovery and success validation are dual. In failure recovery, we are faced with a specific bad outcome that we want to understand and avoid repeating. In success validation, we are faced with some (claimed) good results that we want to validate and replicate or generalize. Both failure recovery and success validation involve understanding causal chains of events and counterfactual possibilities.

More generally, provenance addresses problems of the following form: we have a system description together with some specific inputs and outputs, and we wish to understand (or explain) the process by which the system transformed the inputs to the outputs. Such explanations can have many forms, depending on what aspects of the system behavior are considered critical. Moreover, different kinds of explanation address different informal motivations such as authenticity, integrity, transparency, and accountability. Of course, so far these are all relatively vague concepts that can be interpreted in a number of different ways, and developing more concrete definitions of these concepts is a major part of the challenge.

## 3. Problems with current reality

Provenance is information, and as such, we must consider how to define, manage, and secure it. There are both subjective and objective aspects of provenance that must be carefully untangled. Moreover, we also need to consider semantic issues such as when provenance correctly represents a system or explains a situation. As well, there is the meta-issue of whether provenance information needs to have provenance of its own (and if so, where does this stop?) Satisfying answers to these questions have not yet been found; instead, we perceive a number of basic problems with provenance in today's systems.

***Provenance is incomplete.*** Often this is for efficiency reasons based on dated assumptions about the cost of secondary storage, which is increasingly cheap compared to other hardware. However, there is a more fundamental problem: different applications have different provenance needs, and it is not clear whether there is a plausible "one size fits all" provenance solution that can be hard-wired into general-purpose systems. It may be better to instead develop interfaces for customizing the collection of provenance information.

***Provenance is unreliable.*** Computer systems already manage provenance in numerous ways. Most of these approaches are unreliable in the sense that the guarantees provided are minimal or unclear, and system behavior is unspecified or ad hoc. This places severe limitations on the ability of users to trust distributed data and computations on the Web. If users have mistaken intuitions about the meaning of provenance, they will likely make poor decisions and incorrectly assign blame (or praise). Even worse, if attackers can manipulate provenance information, then users have to consider the possibility of plagiarism, framing and identity fraud. If these risks are well-known, then users will likely simply disregard provenance, leading to wasted collection effort. Having low-quality or easily forged provenance may actually be worse than having no provenance at all!

***Provenance is insecure.*** As discussed by Braun et al. [8], preventing unauthorized access to data is insufficient to ensure that sensitive provenance remains confidential. Conversely, permitting access to (non-confidential) provenance may disclose confidential data. More generally, adding provenance to systems introduces new privacy and security risks even as it increases transparency and accountability, as illustrated for example by the the Iraq intelligence report metadata information leaks in Microsoft Word and other applications. Provenance tracking (like other forms of surveillance) may have a chilling effect on freedom of expression; to avoid this problem, it may be important to preserve "provenance-free zones".

***Provenance is heterogeneous.*** As discussed above, provenance information is currently managed in many different ways for many different purposes in various systems. These

facilities have often grown organically in response to perceived needs, rather than as part of an overall design philosophy. In addition, the information is recorded at many different levels of granularity—whole files vs. individual lines, or whole databases vs. database records. When systems manage provenance in idiosyncratic ways, then it is a major chore to integrate these different kinds of provenance after the fact. Some progress towards a standard format for provenance has been made in the Open Provenance Model effort [25]. This work standardizes a syntax and controlled vocabulary for provenance, but there is still plenty of scope for incompatibility and misinterpretation.

The problem is similar to that of data integration, arising where different databases have been developed to handle similar data but differ in terms of data format, layout, and granularity. Reconciling these changes is highly labor intensive and has motivated (at least!) 20 years of research with no end in sight [34]. Perhaps existing techniques for data integration can be adapted to help with provenance integration. Conversely, provenance information may help attain higher quality in data integration.

***Provenance is non-portable.*** How do we deal with provenance for mobile data that moves among systems? This is a commonplace scenario: consider, for example, curated databases where curators manually browse journal articles, online analysis tools and other databases and copy and paste data into their databases, or any kind of collaboration involving emailing copies of a document back and forth. Recently developed distributed version-control systems, such as Git, Darcs, and Mercurial, may offer some ideas about how to manage provenance for mobile, distributed objects that could be generalized to other settings. However, truly solving this problem may require architectural changes at the hardware, network, operating system, or Web levels (or all of the above).

***Little provenance research is precise, formal or repeatable.*** As argued above, we believe provenance failures are based in part on subjective concerns such as how users interpret metadata. However, we do not believe that this means that we should abandon existing principles for the design, specification, and verification of computer systems. It can be hazardous to implement the first thing that comes to mind without clear understanding of the design space—people might (mis)place faith in a system, with dangerous consequences. Moreover, even if it is ultimately a subjective matter to judge whether a provenance system design is adequate for a given purpose, we should have clear specifications that both users and implementors can rely on.

We believe that both provenance solutions and the problems they are meant to solve need to be formulated clearly in a rigorous framework. Unfortunately, a lot of the work done so far on provenance has fallen short in this respect. Often, the actual or desired behavior of a system (or both) are described only in vague terms. For example, a form of provenance is often motivated as

- being "complete", or "(more) accurate (than another)",
- as capturing information about "(causal) dependences", "influences", "sources", "relevance",
- or as being an "explanation", "justification", or "evidence".

These are loaded terms. They are frequently used as motivation but without any further definition of their meaning in terms of the system that the provenance information is supposed to describe. This is unacceptable because a reader (or user or developer) may interpret one of these terms differently from the way intended by the author, leading to confusion and likely to provenance failures. Moreover, the lack of precise definitions and clear descriptions hinders comparisons in terms of effectiveness, expressiveness or efficiency; even if two systems have similar motivation, their implementations may be so different as to render comparisons meaningless.

Thus, besides our concern with the clarity of descriptions of proposed techniques for provenance, we believe that there are serious gaps in our understanding of the intended goals of such techniques. In terms familiar from computer security, today we see many proposals for mechanisms for provenance, but few clear policies describing what such mechanisms are meant to achieve. Computer security benefited greatly by adopting a formal approach [5], and we believe that more formalism will also be required for provenance technology to succeed.

## 4. Towards solutions

As with many emerging technologies, we expect provenance technology to develop in roughly three stages: research, early adoption, and maturity. We expect provenance to face similar challenges as other *network effect* technologies that become compelling only when a critical mass is reached. However, there are aspects of provenance that seem especially challenging and that we believe don't receive the attention they deserve.

***Semantics*** Many forms of provenance could be captured for a particular system. At a bare minimum, we want to know *where* a particular piece of data comes from [9, 10]. In an e-science setting, however, we additionally want to know *why* an in-silico analysis gave a particular output [10, 15], *how* it was computed [18], or what *dependencies* the output has on different inputs [13]. Some techniques (such as PASS [27] and PASOA [25]) attempt to record everything that a given system component can record, building a detailed trace or "causal graph".

While the above forms of provenance have already been introduced and studied in the literature for *specific* settings and languages—e.g., relational databases, workflows, or

file systems—there is need for generally applicable, formal foundations for provenance. Moreover, since we can only expect provenance to become ubiquitously available if the effort of adding provenance support to systems is relatively low, we need an efficient methodology that allows this general theory to be easily applied to new settings and languages. One issue of current interest is how to integrate provenance models that have been developed separately for databases and workflow systems; more generally, we think it is essential to develop a clearer formal understanding of modularity and abstraction for provenance.

***Traces, incremental, and bidirectional computation*** Many forms of provenance are motivated by a desire to cache intermediate results and support efficient recomputation when the input changes. Incremental recomputation is a classical problem encountered in many different guises throughout computer science (particularly view maintenance in databases [19]). Many approaches employ "traces", or intermediate data structures that elucidate how the output was computed from the input [4]. Thus, when the input is changed, we can propagate the effects forward efficiently by replaying just a part of the trace. Such trace information ought to be computable from a sufficiently rich form of provenance; alternatively, perhaps we should view such traces as a form of provenance in their own right. However, while the "trace everything" approach seems to capture the most provenance-related information, it is currently unclear what language should be used to pose provenance queries on traces. Since such queries essentially inspect program behavior, languages for meta-programming [30] and meta-querying [32] seem like a good starting point.

Provenance has also been linked to the classical *view update problem* in databases [11], and generalizations such as *bidirectional computation* [17]. In recent work on bidirectional transformations, or "lenses", where-provenance information has been found useful for dealing with ordered data collections [6]. In light of these observations, we think that understanding the relationships between provenance, incremental/adaptive computation, and bidirectional computation is a key open problem. For example, can pervasive provenance technology also enable new solutions to these old problems? Conversely, can these existing techniques be used to better understand the foundations of provenance?

***Dependence, information flow, and security*** Information-flow security is based on formalisms such as dependence and noninterference that also underlie program slicing and analysis. Previous work on *dependency provenance* adapted these ideas to motivate a form of provenance that provides a strong guarantee that it captures all dependence information [13].

It is an open question how to define security for provenance. Some previous work (starting with [8, 31]) has considered access control and other security issues for provenance information, but does not formalize security policies or prove correctness of implementations. Chong [14] sets out initial attempts at definitions for security properties for systems in which both data and provenance may be confidential; this work is formulated in terms of detailed operational traces, but seems adaptable to other forms of provenance. However, there are many issues that need further study, such as privacy and audit in the presence of provenance.

***Causality, trust, knowledge, and belief*** These concepts are often invoked as motivations for provenance, but they are nontrivial. For example, the nature of causality has been debated by philosophers for centuries, famously Hume, and continuing to this day. It is far from obvious how to make sense of informal claims that a given provenance record correctly captures a causal relationship, increases trust, or justifies a knowledge or belief, and most research papers don't even try. However, recent work on mathematical models of causality [20], trust [28], and knowledge [16] may provide a good starting point for answering these questions.

## 5. Conclusions

The Web and other technologies are revolutionizing the way we create, share and use information; this revolution offers great benefits but also exposes us to serious new risks. We believe that provenance will play an essential role in this revolution, providing data integrity, trustworthiness, authenticity, and availability, while offering potential benefits to information retrieval, collaboration, and scientific computation.

However, most work on provenance so far has focused on developing systems that address immediate perceived needs, rather than understanding foundational questions such as "what is provenance?", "how do we know when we have enough provenance for a given application?", or "how can we assess or compare approaches to provenance?" We believe that satisfactory answers to these kinds of questions demand serious consideration of underlying issues such as the semantics of provenance and the nature of trust and causality in computer systems. Only once these foundations are better understood—that is, only once the problems we want provenance to solve are clearly defined—can we make real progress on solving them and developing a robust provenance layer for the future Web infrastructure.

## Acknowledgments

This paper is based on discussions started at a series of workshops on provenance in 2008-2009 sponsored by the UK eScience Institute, whose support we gratefully acknowledge.

## References

[1] Principles of provenance wiki. `http://wiki.esi.ac.uk/-Principles_of_Provenance`.

[2] Grand challenges in computing — research. Technical report, British Computing Society, 2004.

[3] INFOSEC hard problem list. Technical report, INFOSEC Research Council, 2005. http://www.infosec-research.org/-docs_public/20051130-IRC-HPL-FINAL.pdf.

[4] Umut A. Acar, Guy E. Blelloch, and Robert Harper. Adaptive functional programming. In *POPL*, pages 247–259. ACM Press, 2002.

[5] David E. Bell and Leonard La Padula. Secure computer system: Unified exposition and Multics interpretation. Technical Report ESD-TR-75-306, MITRE Corp. MTR-2997, Bedford, MA, 1975.

[6] Aaron Bohannon, J. Nathan Foster, Benjamin C. Pierce, Alexandre Pilkiewicz, and Alan Schmitt. Boomerang: resourceful lenses for string data. In *POPL*, pages 407–419. ACM, 2008.

[7] Rajendra Bose and James Frew. Lineage retrieval for scientific data processing: a survey. *ACM Comput. Surv.*, 37(1):1–28, 2005.

[8] Uri Braun, Avraham Shinnar, and Margo Seltzer. Securing provenance. In *HOTSEC'08: Proceedings of the 3rd conference on Hot topics in security*, pages 1–5, Berkeley, CA, USA, 2008. USENIX Association.

[9] Peter Buneman, James Cheney, and Stijn Vansummeren. On the expressiveness of implicit provenance in query and update languages. *ACM Transactions on Database Systems*, 33(4):28, November 2008.

[10] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. Why and where: A characterization of data provenance. In *ICDT*, number 1973 in LNCS, pages 316–330. Springer, 2001.

[11] Peter Buneman, Sanjeev Khanna, and Wang-Chiew Tan. On propagation of deletions and annotations through views. In *PODS*, pages 150–158, 2002.

[12] Susan Carey and Geoffrey Rogow. UAL shares fall as old story surfaces online. Wall Street Journal, September 2008. http://online.wsj.com/article/SB122088673-738010213.html.

[13] James Cheney, Amal Ahmed, and Umut A. Acar. Provenance as dependency analysis. In *DBPL*, volume 4797 of *Lecture Notes in Computer Science*, pages 138–152. Springer, 2007.

[14] Stephen Chong. Towards semantics for provenance security. In *TAPP'09: First workshop on on Theory and practice of provenance*, pages 1–5, Berkeley, CA, USA, 2009. USENIX Association.

[15] Yingwei Cui, Jennifer Widom, and Janet L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM Trans. Database Syst.*, 25(2):179–227, 2000.

[16] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Y. Vardi. *Reasoning About Knowledge*. MIT Press, 2003.

[17] J. Nathan Foster and Grigoris Karvounarakis. Provenance and data synchronization. *IEEE Data Eng. Bull.*, 30(4):13–21, 2007.

[18] Todd J. Green, Gregory Karvounarakis, and Val Tannen. Provenance semirings. In *PODS*, pages 31–40. ACM, 2007.

[19] Ashish Gupta and Iderpal Singh Mumick, editors. *Materialized views: Techniques, Implementations, and Applications*. MIT Press, Cambridge, MA, USA, 1999.

[20] J.Y. Halpern and J. Pearl. Causes and explanations: A structural-model approach–part I: Causes. *British J. Philos. Sci.*, 56:843–887, 2005.

[21] T. Hey and A. E. Trefethen. Cyberinfrastructure for e-science. *Science*, 308(5723):817–821, 2005.

[22] Sandy B. Lewis and William D. Cohan. The economy is still at the brink. New York Times, June 2007. http://www.nytimes.com/2009/06/07/opinion/-07cohanWEB.html.

[23] Clifford A. Lynch. When documents deceive: trust and provenance as new factors for information retrieval in a tangled web. *J. Am. Soc. Inf. Sci. Technol.*, 52(1):12–17, 2001.

[24] Francis L. Macrina. *Scientific Integrity*. ASM Press, 3rd edition, 2005.

[25] Simon Miles, Paul T. Groth, Steve Munroe, Sheng Jiang, Thibaut Assandri, and Luc Moreau. Extracting causal graphs from an open provenance data model. *Concurrency and Computation: Practice and Experience*, 20(5):577–586, 2008.

[26] Greg Miller. A scientist's nightmare: Software problem leads to five retractions. *Science*, 314(5807):1856–1857, December 2006.

[27] Kiran-Kumar Muniswamy-Reddy, David A. Holland, Uri Braun, and Margo Seltzer. Provenance-aware storage systems. In *USENIX Annual Technical Conference*, pages 43–56. USENIX, June 2006.

[28] Jordi Sabater and Carles Sierra. Review on computational trust and reputation models. *Artif. Intell. Rev.*, 24(1):33–60, 2005.

[29] Yogesh Simmhan, Beth Plale, and Dennis Gannon. A survey of data provenance in e-science. *SIGMOD Record*, 34(3):31–36, 2005.

[30] Walid Taha and Tim Sheard. MetaML and multi-stage programming with explicit annotations. *Theoretical Computer Science*, 248(1–2):211–242, 2000.

[31] Victor Tan, Paul T. Groth, Simon Miles, Sheng Jiang, Steve Munroe, Sofia Tsasakou, and Luc Moreau. Security issues in a SOA-based provenance system. In Luc Moreau and Ian T. Foster, editors, *IPAW*, volume 4145 of *Lecture Notes in Computer Science*, pages 203–211. Springer, 2006.

[32] Jan Van den Bussche, Dirk Van Gucht, and Gottfried Vossen. Reflective programming in the relational algebra. *Journal of Computer and System Sciences*, 52(3):537–549, 1996.

[33] Sam Varghese. UK government gets bitten by Microsoft Word. Sydney Morning Herald, July 2003. http://www.smh.com.au/articles/2003/07/02/-1056825430340.html.

[34] Patrick Ziegler and Klaus R. Dittrich. Three decades of data integration - all problems solved? In René Jacquart, editor, *IFIP Congress Topical Sessions*, pages 3–12. Kluwer, 2004.