What Is the Best Thing to Do Next?
A Tutorial on Intensional Data Management

Antoine Amarilli
Institut Mines–Télécom;
Télécom ParisTech; CNRS LTCI
Paris, France

Pierre Senellart
Institut Mines–Télécom; Télécom ParisTech;
CNRS LTCI & NUS; CNRS IPAL
Paris, France & Singapore

firstname.lastname@telecom-paristech.fr

ABSTRACT

We call data *intensional* when it is not directly available, but must be accessed through a costly interface. Intensional data naturally arises in a number of data management scenarios, such as crowdsourcing, Web crawling, or ontology-based data access. Such scenarios require us to model an uncertain view of the world, for which, given a query, we must answer the question "What is the best thing to do next?" Once data has been retrieved, the knowledge of the world is revised. This tutorial is an introduction to intensional data management, with a review of the solutions brought in various areas of data management and machine learning, and of some challenging open problems.

1. INTRODUCTION

Intensional Data Management. Many data-centric applications involve data that is not directly available in extension, but can only be obtained after some access to the data is made, at some form of cost. In traditional database querying [13], the access may be disk I/O, and the I/O cost will depend on which indexes are available. In crowdsourcing platforms [4, 25], accessing data involves recruiting a worker to provide the data, and the cost is in terms of monetary compensation for workers and latency to obtain the data. In Web crawling [16], accesses are HTTP requests and cost involves bandwidth usage, network latency, and quota use for rate-limited interfaces. In ontology-based data access [10], accesses mean applying a reasoning rule of an ontology, and the cost is the computational cost of such an evaluation.

We abstract out the general problem of accessing data through costly interfaces as that of *intensional data management*. This terminology contrasts with *extensional data management* where data is freely accessible (entirely stored in-memory, or locally stored on disk in situations when disk accesses are negligible). The terminology is in line with that of Datalog [2], where intensional data is data not initially present but obtained through rule applications; it is also the terminology used in Active XML [1]. *Intensional data* is sometimes used to refer to the schema of a database, as opposed to extensional facts, especially in the setting of deductive databases [28]; in the same way, in intensional data management, we study how to perform query optimization and other data management tasks when only the schema (and access methods) to some of the data is directly available, not the facts.

Intensional data management applications share a number of distinguishing features. At every point in time, one has an *uncertain view of the world*, that includes all the data that has already been accessed, together with the schema, access methods, and some *priors* about what data remain to be accessed. Given a user's query, the central question in intensional data management is: "What is the best thing to do next" in order to answer the query, meaning, what is the best access that should be performed at this point, given its cost, potential gain, and the uncertain knowledge of the world. Once an access is chosen and performed, some data is retrieved, and the uncertain view of the world must be revised in light of the new knowledge obtained. The process is repeated until the user's query receives a satisfactory answer or some other termination condition is met.

Use Cases. To illustrate, let us give some concrete examples of complex use cases involving intensional data management.

Consider the application of *mobility in smart cities*, i.e., a system integrating information about transportation options, travel habits, traffic, etc., in and around a city. Various public resources can be used to collect and enrich data related to this application: the Web, deep Web sources, social networking sites, the Semantic Web, annotators and wrapper induction systems, crowdsourcing platforms, etc. Moreover, in such a setting, domain-specific resources, not necessarily public, contribute to the available data: street cameras, red light sensors, air pollution monitoring systems, etc. Users of the system, namely, transport engineers, ordinary citizens, etc., may have many kinds of needs. They can be simple queries expressed in a classical query language (e.g., "How many cars went through this road during that day?" or "What is the optimal way to go from this place to that place at a given time of day?"), certain patterns to mine from the data ("Find an association rule of the form $X \Rightarrow Y$ that holds among people commuting to this district."), or higher-level business intelligence queries ("Find anything interesting about the use of the local bike rental system in the past week.").

To be very concrete, let us imagine what options are available for the query "How many cars went through this road during that day?". One could:

- If applicable, use data from electronic toll gates;
- Use a computer vision program to analyze the street camera feeds and automatically extract each passage of a vehicle;
- Ask crowd workers to perform the same analysis;
- Do the same, but only a fraction of the day, and extrapolate the results;

- Use traffic data from Bing Maps API, correlated with external data about road characteristics;
- Analyze the mentions of this road segment on social media, to determine both its usage rate, and a subjective idea of it being overloaded;
- Send a team of expert traffic specialists to survey the road;
- etc.

Each of these (and each combination of these) has a cost (in terms of manpower, budget, processing time, bandwidth) and a precision. The objective is to obtain an optimal solution given a precision threshold. This example was fairly simple, but imagine that determining the traffic on a road may be just one component of a more complex information need, such as redesigning an entire district.

As a second example, consider the problem of *personal information management*, namely, integrating user data across services that manage the user's emails, calendar, social network, travel information, etc. To answer a query such as "find the people I need to warn about my upcoming trips", the system would have to orchestrate queries to the various services: extract the trips, identify the meetings that conflict with them, and determine their likely participants.

A third example is *socially-driven Web archives* [29]: their goal is to build semantically annotated Web archives on specific topics or events (investment for growth in Europe, the 2014 Winter Olympics, etc.), guiding the process with clues from the social Web as to which documents are relevant. These archives can then be semantically queried by journalists today or historians tomorrow, e.g., to retrieve all resources mentioning a given person. The construction of these archives relies on Web crawling, deep Web harvesting, access to social networking sites such as Twitter or YouTube via their APIs, use of tools for information extraction, named entity recognition, opinion mining, etc. Again, these various intensional sources come with very different costs, and an optimal plan for a user's query involve choosing an optimal way to collect, annotate, and query the different relevant sources.

Tutorial Content. This tutorial covers the general field of intensional data management by defining it as an abstract problem, giving concrete instances of it in the form of the previous use cases, and then, as we shall briefly do in Section 2, presenting the solutions to components of the intensional data management problem that have been proposed in very different areas of the database and machine learning literature: crowdsourcing, Web crawling, adaptive query evaluation, answering queries using views, querying under access limitations, reinforcement learning, active learning, etc. We will also discuss some of the most challenging open issues in intensional data management (see Section 3). Practical issues about the tutorial organization are discussed in Section 4.

2. MAIN APPROACH

We now present the general steps in intensional data management, and describe which areas of research have been concerned with each step and what solutions they have brought. A summary of the general approach is presented in Figure 1, with references to corresponding sections.

First, we must design a model to represent the state of the world, and more generally our uncertainty about its state. Next, we must deal with the core of the problem: decide which access allows us to make the most progress on our query, as a function of our current knowledge and the possible accesses on the various sources. The third step is to update our knowledge of the world with the results of the chosen access.

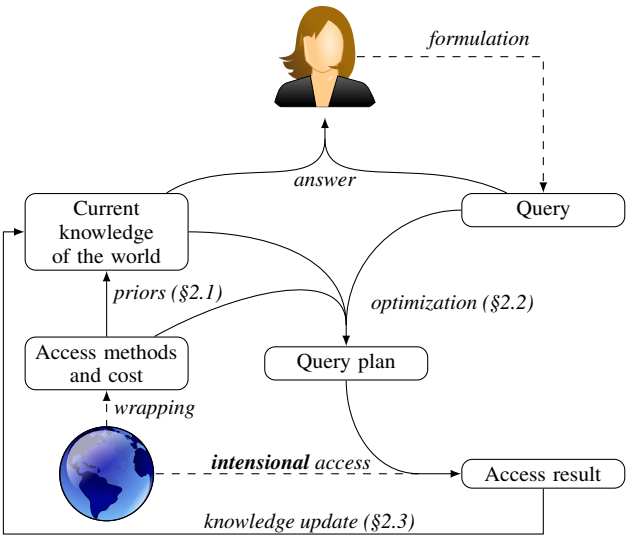

Figure 1: Approach to Intensional Data Management

2.1 Modeling the World

At any point in time, the system must have a representation of its current state. In all generality, the state is a description, constructed from the past observations and prior knowledge, of a *probability distribution* on all *possible worlds*, that is, all possible states of the world which we access through the sources. Of course, this representation must be *concise*, as we cannot write out explicitly the (possibly infinite) collection of possible worlds; it must also be *operational*, in that we must have an efficient way to perform the next steps of the approach on it: deciding which access to perform next, integrating results of the chosen access, but also determining our current best answer for the query.

Existing approaches for such representations can be distinguished first and foremost by the *kind of data* which they attempt to represent. For simple, unstructured data, such as the answers to a binary question in a crowdsourcing application, the representation may just be a choice of parameters fitted to the distribution of answers. More interestingly, for structured data, we can turn to probabilistic representations for relational databases [32] and XML [19] data management systems: those frameworks annotate relational or XML instances with information about the uncertainty of the data items.

Second, we can distinguish probabilistic representations between *open-world* and *closed-world*, depending on whether the set of possible worlds is finite or infinite. A third distinction is on whether the representations are only about *incompleteness* ("what are the possible worlds") or whether they extend to proper *probability distributions* on them. As an example, open-world incomplete representations are often used for *open-world query answering*, where the possible worlds are all the completions, subject to known logical constraints, of the facts that are known to hold. The constraints are generally deterministic, although some works [15] allow uncertain rules (by reduction to deterministic rules). Open-world probabilistic representations are rare, because of modeling issues for infinite probabilistic spaces. One important exception is [9], which presents an open-world probabilistic framework for XML documents via recursive Markov chains.

2.2 Choosing Accesses

Once we have represented our uncertain knowledge about the world, the main problem is to choose which access to perform next.

Of course, this choice depends on the kind of accesses that are possible. For instance, in crowdsourcing contexts, the simplest setting is when the possible questions to ask deal with the classification of *separate* items [34]: in this case, the choice reduces to determining on which item we want to make progress. By contrast, in crowd situations where the answers depend on each other, the problems becomes more complex [3].

More generally, the possible accesses can be different "views" on the same information. In the setting of Web sources, the language of *binding patterns* [27] is used to represent the various possibilities (with different restrictions) to access information about the same relations. Even more generally, in the context of data integration [21], or of query answering using views [17], there can be very general dependencies between the views (that are accessed) and the underlying data (on which the query is posed): e.g., equality-generating or tuple-generating dependencies. In such general situations, it can become undecidable to determine, under arbitrary dependencies, whether a given access is relevant to the query [7, 8].

Having fixed our representation of the accesses, we must now decide which one to perform, which we call the *access choice problem*. A simple idea would be to compile or rewrite our query to the accesses, in a *static* manner, and then execute this plan; however, we are interested here in an *interactive* approach, where each access is chosen depending on the result of the previous accesses. This dichotomy can be seen in crowdsourcing, between some works [25] that prepare batches of queries executed independently of previous answers, and others [24] that make interactive decisions to decide when to stop.

The same dichotomy can distinguish approaches for query evaluation (directly, or via views), with the static approach being that of evaluating a fixed plan (or a fixed rewriting) for the query [13], and the interactive approach being known as *adaptive query evaluation* [11] . This line of work deals with query evaluation plans that *adapt* depending on the actual performance of the query being evaluated: for instance, by adding query optimization operators, or trying out several plans in parallel on subsets of the data. A last example of another access choice problem with interactive solutions is that of *focused crawling*, to locate interesting information by choosing which pages to query [22], or *deep Web crawling*, applying the same idea to information located behind Web forms [23].

Another angle to see the access choice problem is that of machine learning, namely *reinforcement learning* and *active learning*. Reinforcement learning [5, 26, 33] is the study of how to maximize rewards in the following setting: whenever we find ourselves in a state, we can choose an action to perform, which yields a reward and changes the state. This implies an inherent tradeoff between exploration (trying out new actions leading to new states and to potentially high rewards) and exploitation (performing actions already known to yield high rewards). We can model the access choice problem as a reinforcement learning problem on a huge structured state space corresponding to the possible states of our intermediate knowledge, as proposed in [6] for data cleaning.

Active learning [30] deals with the problem of optimally using an oracle, which is costly to access, to provide labels for training data that will be used to build a learning model, e.g., a classifier. This implies a tradeoff between the cost of oracle calls, and the cost of errors on the task. Active learning can be seen as an access choice problem between two accesses: one which is costly but certain, and another which is a cheap but noisy extrapolation based on the current knowledge.

2.3 Updating the Representation

Once an access has been performed and results have been obtained, the representation of the world must be updated accordingly. A general framework to perform this task is that of *Bayesian inference* [14]: integrate the observation to our prior knowledge to obtain a posterior representation of the world.

Updates are especially hard to perform in the common situation when the access only returns uncertain information about the actual data. For instance, say an access gave us the *number* of data items of interest (which may be useful, e.g., to decide to retrieve them one by one or in bulk); we must represent the existence of these items, though we know nothing about them. This problem also occurs in crowdsourcing contexts: constraints on access results may force us to extrapolate additional information [3].

Updating probabilistic representations has been studied for probabilistic XML documents [18], and as *conditioning* [20] for probabilistic databases: restrict the possible worlds of a database with an additional constraint (a logical rule, or, here, an observation). In most situations, however, this task is intractable.

3. OPEN PROBLEMS

The intensionality of data, but also the heterogeneity of its structure, and our uncertainty about the true state of the world, are three major challenges of intensional data management. What is more, these challenges are not independent, but interact tightly. For example, if we use a probabilistic modeling of uncertainty, we need to represent, manage, query, probability distributions on structured objects, so that the representation of uncertainty depends on the structure that we use. Likewise, the kinds of intensional accesses which we may perform depend on the structure of the data considered, and structural constraints can be used to restrict the kind of accesses to make. Last, our intensional accesses will depend on our representation of uncertainty, as this representation may be used, e.g., to *predict* the results of accesses which have not been performed yet. Hence, we simply cannot use independent solutions to address each challenge.

Another difficulty is the intractability of many of the subproblems that are tackled. Conditioning a probabilistic database by some logical constraint [20], optimally crawling a deep Web site [16], determining whether an access is relevant to a query [8], or even merely querying a probabilistic database [32] are all intractable problems (NP-hard or beyond), even in simple enough settings. Hence, on the one hand we must find simplified problems for which it is still tractable to find an optimal iterative plan, and on the other hand we must devise heuristic and approximate strategies that are not guaranteed to be optimal (but whose error should be bounded).

Most existing approaches assume a uniform notion of cost. In reality, we must evaluate cost along multiple heterogeneous dimensions: money, CPU time, bandwidth limits, policy constraints, etc. For this reason, in contrast to traditional query optimization, we cannot use a single value to model cost, and multi-objective optimization is required. Another challenge is that though in some cases the cost may be known in advance, in others (e.g., when the cost is computation time) it can only be observed after the fact, and used to infer the cost of similar future data accesses. This introduces another level of uncertainty when modeling sources. Note that some works in the active learning field have started exploring more realistic notions of cost [12, 31].

Active learning [30] and reinforcement learning [33] are powerful tools to decide the next action to perform. However, in contrast with active learning, in intensional data management we discover the data as we access it: there is no fixed set of data points with known features to choose from, but there are structural constraints on the data. Similarly, reinforcement learning assumes a fairly simple data model, e.g., the independent states of Markov decision

processes [26]. In reality, states have a complex structure, namely that of the data. To benefit from the vast literature on optimizing rewards in reinforcement learning, and optimally choosing accesses in active learning, we first need to integrate into these models the support of structure and logical constraints.

4. ABOUT THE TUTORIAL

Tutorial format. This tutorial proposal is for a 1.5 hour tutorial that will review the various fields of the scientific literature dealing with intensional data management, as highlighted in the previous sections. A special focus will be put on connections between these areas, and on the description of open problems. Upon request, an extension to a 3 hour tutorial is possible and will allow us to cover specific fields in more depth, but we feel the audience would be more interested in a high-level view of the different areas and how techniques from one field can be applied to another, rather than in an in-depth review of these areas.

The tutorial proposal is fully novel and the material covered, though obviously described in the literature, has never been presented in this integrated form, as far as we know. Though there have been various tutorials on uncertain data management or on crowdsourcing in database conference, neither is our focus here. The tutorial will be accessible to a general audience of researchers in data management, data mining, or machine learning, both from systems and theory backgrounds.

Presenters' biographies. Antoine Amarilli is a PhD candidate at Télécom ParisTech supervised by Pierre Senellart. He obtained his MSc from École normale supérieure in 2012. His PhD research is on the theoretical aspects of the management of uncertain, structured, and intensional data. See `http://a3nm.net/`.

Pierre Senellart is a Professor of Computer Science at Télécom ParisTech, and a Senior Research Fellow at the National University of Singapore, within the IPAL laboratory. He obtained his PhD from Université Paris–Sud in 2007. His research interests focus around practical and theoretical aspects of Web data management, including Web crawling and archiving, Web information extraction, uncertainty management, Web mining, and querying under access limitations. See `http://pierre.senellart.com/`.

5. REFERENCES

[1] S. Abiteboul, O. Benjelloun, B. Cautis, I. Manolescu, T. Milo, and N. Preda. Lazy query evaluation for Active XML. In *SIGMOD*, 2004.

[2] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[3] A. Amarilli, Y. Amsterdamer, and T. Milo. Uncertainty in crowd data sourcing under structural constraints. In *UnCrowd*, 2014.

[4] Y. Amsterdamer, Y. Grossman, T. Milo, and P. Senellart. Crowd mining. In *SIGMOD*, 2013.

[5] J.-Y. Audibert, R. Munos, and C. Szepesvári. Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theor. Comput. Sci.*, 410(19), 2009.

[6] M. Benedikt, P. Bohannon, and G. Bruns. Data cleaning for decision support. In *CleanDB*, 2006.

[7] M. Benedikt, P. Bourhis, and P. Senellart. Monadic Datalog containment. In *ICALP*, 2012.

[8] M. Benedikt, G. Gottlob, and P. Senellart. Determining relevance of accesses at runtime. In *PODS*, 2011.

[9] M. Benedikt, E. Kharlamov, D. Olteanu, and P. Senellart. Probabilistic XML via Markov chains. *PVLDB*, 3(1), 2010.

[10] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, and D. F. Savo. The MASTRO system for ontology-based data access. *Semantic Web*, 2(1), 2011.

[11] A. Deshpande, Z. G. Ives, and V. Raman. Adaptive query processing. *Foundations and Trends in Databases*, 1(1), 2007.

[12] P. Donmez and J. G. Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *CIKM*, 2008.

[13] H. Garcia-Molina, J. D. Ullman, and J. Widom. *Database systems - the complete book (2. ed.)*. Pearson Education, 2009.

[14] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian Data Analysis*. 3 edition, 2013.

[15] G. Gottlob, T. Lukasiewicz, M. V. Martinez, and G. I. Simari. Query answering under probabilistic uncertainty in Datalog+/- ontologies. *Ann. Math. Artif. Intell.*, 69(1), 2013.

[16] G. Gouriten, S. Maniu, and P. Senellart. Scalable, generic, and adaptive systems for focused crawling. In *Hypertext*, Sept. 2014.

[17] A. Y. Halevy. Answering queries using views: A survey. *VLDBJ*, 10(4), 2001.

[18] E. Kharlamov, W. Nutt, and P. Senellart. Updating probabilistic XML. In *Updates in XML*, 2010.

[19] B. Kimelfeld and P. Senellart. Probabilistic XML: Models and complexity. In *Advances in Probabilistic Databases for Uncertain Information Management*. Springer, 2013.

[20] C. Koch and D. Olteanu. Conditioning probabilistic databases. *PVLDB*, 1(1), 2008.

[21] M. Lenzerini. Data integration: A theoretical perspective. In *PODS*, 2002.

[22] F. Menczer, G. Pant, P. Srinivasan, and M. E. Ruiz. Evaluating topic-driven web crawlers. In *SIGIR*, 2001.

[23] R. Nayak, P. Senellart, F. M. Suchanek, and A. Varde. Discovering interesting information with advances in Web technology. *SIGKDD Explorations*, 14(2), 2012.

[24] A. G. Parameswaran, H. Garcia-Molina, H. Park, N. Polyzotis, A. Ramesh, and J. Widom. Crowdscreen: algorithms for filtering data with humans. In *SIGMOD*, 2012.

[25] A. G. Parameswaran, A. D. Sarma, H. Garcia-Molina, N. Polyzotis, and J. Widom. Human-assisted graph search: it's okay to ask questions. *PVLDB*, 4(5), 2011.

[26] M. L. Puteman. *Markov Decision Processes*. Wiley, 2005.

[27] A. Rajaraman, Y. Sagiv, and J. D. Ullman. Answering queries using templates with binding patterns. In *SIGMOD*, 1995.

[28] R. Reiter. Deductive question-answering on relational data bases. In *Logic and Data Bases*, 1977.

[29] T. Risse, S. Dietze, W. Peters, K. Doka, Y. Stavrakas, and P. Senellart. Exploiting the social and semantic Web for guided Web archiving. In *TPDL*, 2012. Poster.

[30] B. Settles. *Active Learning*. Morgan & Claypool Publishers, 2012.

[31] B. Settles, M. Craven, and L. Friedland. Active learning with real annotation costs. In *Cost-Sensitive Learning*, 2008.

[32] D. Suciu, D. Olteanu, C. Ré, and C. Koch. *Probabilistic Databases*. Morgan & Claypool Publishers, 2011.

[33] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, 1998.

[34] X. Yang, R. Cheng, L. Mo, B. Kao, and D. Cheung. On incentive-based tagging. In *ICDE*, 2013.