Management of Data with Uncertainties*

Christopher Re
University of Washington
Seattle, WA
chrisre@cs.washington.edu

Dan Suciu
University of Washington
Seattle, WA
suciu@cs.washington.edu

Categories and Subject Descriptors

F.4.1 [**Mathematical Logic**]; G.3 [**Probability and statistics**]; H.2.5 [**Heterogeneous databases**]

General Terms

Algorithms, Management, Theory

Keywords

Probabilistic databases, Query processing

Since their invention in the early 70s, relational databases have been deterministic. They were designed to support applications s.a. accounting, inventory, customer care, and manufacturing, and these applications require a precise semantics. Thus, database systems are deterministic. A row is either in the database or is not; a tuple is either in the query answer or is not. The foundations of query processing and the tools that exists today for managing data rely fundamentally on the assumption that the data is deterministic.

Increasingly, today we need to manage data that is uncertain. The uncertainty can be in the data itself, in the schema, in the mapping between different data instances, or in the user query. We find increasingly large amounts of uncertain data in a variety of domains: in data integration, in scientific data, in information extracted automatically from text, in data from the physical world. Large enterprises today can sometimes afford to cope with the uncertainty in their data by completely removing it, by using some expensive data cleaning or ETL tools. But increasingly today organizations or users need to cope directly with uncertain data, either because cleaning it is prohibitively expensive (e.g. in scientific data integration or in integration of Web data), or because it is even impossible to clean (e.g. sensor data or RFID data). It becomes clear that we need to be able to manage data that is uncertain. Such data is usually represented by explicitly annotating it with the degree of uncertainty, which is almost always a probability. Hence, our quest to develop data management techniques for probabilistic data. In most cases only a fraction of a large database is uncertain: it can be a table representing the fuzzy match between two otherwise clean data sources, or some automatically extracted information enriching a clean, reference database. The tools that we develop for managing uncertain data must therefore be natural extensions of the current database management tools.

Probabilistic databases have been studied almost continuously since the late 80's [10, 6, 28, 41, 60]. Recently, there has been an increased interested in probabilistic databases and in the management of probabilistic data in general because of the need to cope with uncertainties in large scale data management. Active research groups working on probabilistic data management include, among others, the Trio project at Stanford [58], the MystiQ project at the University of Washington [9, 48], the Orion project at Purdue University [47, 12], the group at the University of Maryland [31], and the MayBMS project at the University of Saarland [4]. Far from being solved, the problem of probabilistic data management is an active and challenging research area, spanning data management, probabilistic inference, and logic. In this short paper we illustrate a few scenarios requiring the management of uncertain data, then give a brief overview of the main research questions in the area.

1. INSTANCES OF UNCERTAIN DATA

In *fuzzy object matching* the problem is to reconcile objects from two collections that have used different naming conventions. This is a central problem in data cleaning and data integration, and has also been called record linkage, deduplication, or merge-purge [26, 3, 11, 29, 34, 36, 14, 59, 5]. The basic approach is to compute a similarity score between pairs of objects, usually by first computing string similarity scores on their attributes, then combining these scores into a global score for the pair of objects. Next, this score is compared with a threshold, and each pair of objects is classified into a *match*, a *non-match*, and a *maybe-match* [3]. Ré et al. [48] propose to convert these scores into probabilities[1] and store them directly in a probabilistic database. There is no more need for a threshold: instead all potential matches are stored together with their probabilities, then used during query processing to rank the answers. Fig 1 illustrates

*This research was supported in part by NSF grants IIS-0415193, IIS-0627585, IIS-0513877, IIS-0428168, and a gift from Microsoft.

CIKM'07, November 6–8, 2007, Lisboa, Portugal.

[1]They are often already expressed as a probability.

MovieReviewMatch

Review	Movie	P
12 Monkeys	Twelve Monkeys	0.4
12 Monkeys	Twelve Monkeys (1995)	0.3
12 Monkeys	Monk	0.013
Monkey Love	Twelve Monkeys	0.35
Monkey Love	Love Story	0.27

This example is adapted from [48]

Figure 1: Illustration of a fuzzy join between some movies in the IMDB movie database and reviews found on the Web. The movie titles and the reviews almost never match exactly. The fuzzy join computes a similarity score between the movie titles and the reviews. The result is a probabilistic table.

Text document: `...52 A Goregaon West Mumbai ...`

PersonAddress

PersonID	House-No	Area	City	P
1	52	Goregaon West	Mumbai	0.1
1	52-A	Goregaon West	Mumbai	0.5
1	52	Goregaon	West Mumbai	0.2
1	52-A	Goregaon	West Mumbai	0.3
2	7	Westlake	...	...

This example is adapted from [35].

Figure 2: Text segmentation of street addresses. Each address has several possible segmentations. Wile the standard approach is to retain only the most likely segmentation, the other possible segmentations are often quite important in query answering, and can be kept in a probabilistic relation.

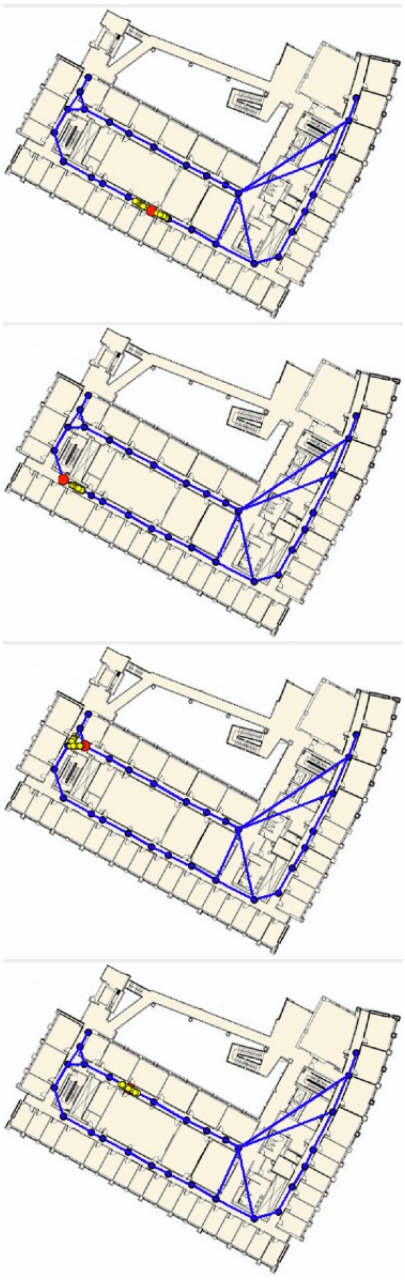
a small fragment of such a match table between movie titles from IMDB and reviews from the Web.

The goal of *Information Extraction* is to extract structured data from a collection of unstructured text documents. Usually the schema is given in advance by the user, and the extractor is tailored to that specific schema [51]. All approaches to extraction are imprecise, and most often can associate a probability score to item extracted. Gupta and Sarawagi [35] describe how to use a probabilistic database to store the result of text segmentation with Conditional Random Fields: Fig. 2 illustrates the possible segmentations of an address, together with their probabilities.

Data produced by information extraction systems is often used by applications in conjunction with other data. For example a marketing company may integrate this data with a data about incomes to answer the query *find 500 people with the highest income living in West Mumbai*. Correct extractions are critical for such applications otherwise the query has low recall, e.g. it will miss all customers whose real address is *West Mumbai* but the system extracted *Mumbai* instead. Today's common practice is to retain only the most likely extraction for each piece of text, and discard the others, but this loses a lot of valuable information. Indeed [35] has shown a strong correlation between the probability returned by a CRF-based extractor and the fraction of extractions that are correct. For example if one manually inspects all extracted items that have probability between, say, 0.3 and 0.35, then one finds that approximatively $30 - 35\%$ of them are correct in a given instance. As a consequence, keeping only the most likely segmentation will miss many correct segmentations. Instead, [35] proposes to keep all segmentations in the probabilistic table, hence the need to support general-purpose queries on a probabilistic database.

RFID Data Management is an emerging class of applications whose goal is to manage readings of RFID tags. The *RFID Ecosystem* deployed at the University of Washington is based on a building-wide RFID infrastructure with 80 RFID readers, 300 antennas, tens of tagged people, and thousands of tagged objects [55]. All data is streamed from the readers into a central database, where it is accessible to applications. One goal of the RFID Ecosystem is to study and develop applications of RFID-based, community-oriented pervasive computing, and the data management aspects play a central role in this study. The raw data collected by the RFID system is a table `Sighting(Time, TagID, AntennaID)`, where each triple captures the time and location where a reader sighted an RFID tag near one of its antennas. Four such readings are illustrated by the large (red) dots in the four diagrams in Fig. 3. The raw antenna readings are too noisy to be used directly by the applications: readings are missed, or tags are read by two adjacent antennas, etc. Probabilistic models are used to extract higher level events from the raw, low level readings. For example, as a first step, a particle filter is used to compute a probability distribution on a person's location, as that person walks along a corridor: these are the small, yellow dots in Fig. 3. The resulting data is thus probabilistic, and is stored in a probabilistic table `LocatedAt(Time, TagID, Location, P)`, as shown in Fig. 3. The next step is to develop applications that query this data in complex way, e.g. *retrieve all students who took my book to the coffee room*, and for that one needs a probabilistic database management system.

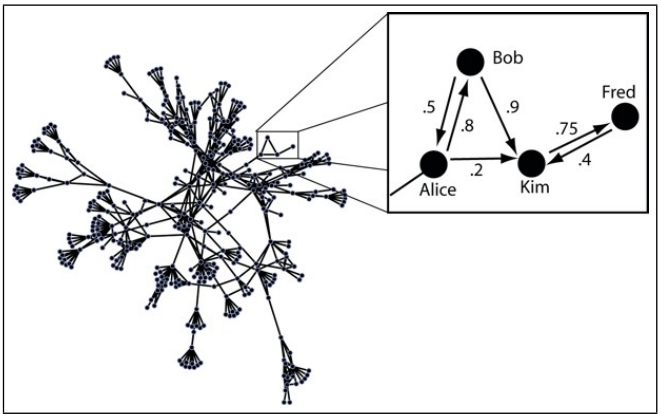
Social Network Analysis A new and interesting example of uncertain data are social networks [2]. The data in large social networks is invaluable for marketing, health, communication, and other applications, and this has renewed the interest in Social Network Analysis (SNA) [27, 54]. Novel social network applications enable individuals to connect with old friends and colleagues and form bridges to new individuals in areas ranging from business (e.g. Visible Path [45] and Linked In [39]) to socialization (e.g. Facebook [25] and MySpace [44]) and to entertainment (e.g. iLike [15]). Given the way the data about social networks is collected, it is almost always imprecise. For example, data is collected through automated sensors [13], or is anonymized communication data (e.g. e-mail headers [1]), or is extracted from self-reporting/logging on Internet-scale networks [17, 30] as a proxy for real relationships and interactions. Furthermore, approximation algorithms [57] intended to calculate network properties (e.g. various centrality measures) on these increasingly large networks creates additional uncertainty. Typically the data for SNA is a graphical representation in which nodes—called *actors* —represent individuals or groups. An edge (potentially labeled) in this graph represents the relationship between actors and generally indicates the possibility of information flow between them. In the early history of SNA, this graph data was collected by survey, interview, and other observational techniques [27, 37, 54, 56]. While the results were potentially tainted by biased observations, missed observations, and misreporting, the intimate involvement of the researcher (frequently, over extended periods) provided some confidence that the data



LocatedAt

Time	TagID	Location	P
1244	T388	L2	0.2
		L3	0.4
		L5	0.4
1245	T388	L3	0.3
		L4	0.1
		L5	0.2
		L6	0.4
1246	...		

Figure 3: An example in which raw readings from RFID tags are converted into a probabilistic table representing a person's location. The RFID antennas are the large blue dots: the red dots in each diagram show which antenna is receiving a signal from the RFID tag at that time. A particle filter generates at each moment in time 100 samples of the persons location, which can be used to derive a probability space over that person's location. This is stored in a probabilistic table illustrated here. The images are courtesy of Julie Letchner.



Influences

Name1	Name2	P
Alice	Bob	0.5
Alice	Kim	0.2
Bob	Kim	0.9
Bob	Alice	0.5
Kim	Fred	0.75
Fred	Kim	0.4

This example is adapted from [2]

Figure 4: A social network

was precise. As those studying and utilizing social networks have moved to enormous scales, they have frequently sacrificed some accuracy as careful methodologies have become increasingly difficult or impossible. Furthermore, in wild and uncontrolled environments such as the Internet, biases can develop due to application design (e.g. default friends on MySpace) and malicious individuals (e.g. spammers building network connections in some automated way). The result of this "noise" is the introduction of tremendous levels of uncertainty in the data which are ill-supported by current large scale data management systems. A brief illustration of a social network and its representation as a probabilistic database is given in Fig. 4.

The quest for a general purpose probabilistic database management system. Clearly, what all these applications have in common is their need to manage some probabilistic data. In addition, they need to support general-purpose queries over this probabilistic data, or over a mixed database consisting of both probabilistic and deterministic databases. Consider for example a query asking for all movies that received a high review from both reviewers Jim and Joe (a *two thumbs up* movie):

```
SELECT DISTINCT m.title
FROM Movie x, MovieReviewMatch m1, Review y1,
             MovieReviewMatch m2, Review y2
WHERE x.title = m1.movie and m1.review = y1.review
  and y1.reviewer = 'Joe' and y1.rating >= 3
  and x.title = m2.movie and m2.review = y2.review
  and y2.reviewer = 'Jim' and y2.rating >= 3
```

The query mentions twice the table `MovieReivewMatch` in Fig. 1 in addition to two deterministic tables, `Movie` and `Review`. The challenge for the query processor is to compute not just the distinct movie titles, but also their probabilities, based on the uncertain evidence contained in the

MoviewReviewMatch table. Moreover, it needs to integrate this probabilistic inference with standard query processing, s.a. index lookups, join computations, and groupbys. The answers to such a query will be returned to the user annotated with the systems' confidence (a probability), and ranked in decreasing order of that confidence. It is critical for all applications of uncertain data to support such general-purpose queries, expressed in some database query language s.a. SQL.

2. CHALLENGES

Query processing The answer to a standard SQL query over a probabilistic database is a set of probabilistic tuple answers: each tuple returned by the system has a probability of being in the query's answer set. The system needs to compute these probabilities, and is this difficult, because it is an instance of the *probabilistic inference problem*, which has been extensively studied in the AI literature [16], and is known to be notoriously hard [40]. To process large scale probabilistic data we need to develop specific probabilistic inference techniques that can be integrated well with SQL query processors and optimizers, and that scale to large volumes of data. This has a lead to a line of research [20, 19, 52, 48] that has adapted probabilistic inference algorithms to various classes of SQL queries and various representation formalisms for probabilistic data. An important result from this line of research is the observation that database queries admit a dichotomy in terms of their complexity over probabilistic databases: each query q can either be computed in PTIME in the size of the probabilistic database, or is #P-hard [2] in the size of the probabilistic database. The complexity of the query evaluation problem on probabilistic databases has been studied in [32, 20, 23, 22, 49] for various SQL fragments and various representation formalisms.

Representation formalisms In its most general form, a probabilistic database is a probability space over all possible instances of the database, called *possible worlds*. It is impossible in practice to enumerate all possible instance: instead we need a concise representation formalism that can describe all possible worlds and their probabilities. The most commonly used techniques in Knowledge Representation exploit conditional independence between variables and represent a probability space in terms of a graphical model [46]. In probabilistic databases, because they are used to manage data with uncertainties, we also need to represent the *lineage*, or the *provenance* of each item in the data, to capture the reason for its uncertainty. The problem of representing both the uncertainty and the lineage has been addressed in the Trio project [58, 8, 7], which has introduced the term ULDB. An important observation of this line of research is that the lineage can be expressed naturally using some form of boolean expressions. Such expressions had been studied before to represent incomplete databases (c-table) [38] and probabilistic databases [28]. In particular, the lineage contains sufficient information to enable a general-purpose probabilistic inference algorithm to compute the output probabilities of any SQL query, although using this approach in practice is likely to be quite inefficient. More generally, lineage and provenance expressions are instances of expressions in a commutative semiring, as described by Green at al. [33].

[2] #P is a complexity class introduced by Valiant [53] and consists of all numerical functions f s.t. there exists a polynomial time, non-deterministic Turing Machine M s.t. on every x, $f(x)$ is the number of accepting computations of M on input x.

Query answering from probabilistic views In many scenarios we do not have access to the full information about the probability space. One example is the partial representation of probabilistic views described in [50]. Here the representation only describes which tuples are independent and which tuples are disjoint, leaving unspecified all other correlations. This representation is very efficient, since it removes a lot of detailed lineage information, but a query can use the view only if its answer is independent on the correlations left unspecified. This enables a system to use previously answered queries to speedup the evaluation of a new query, much in the spirit of query answering using views and semantic caching. In another, more general scenario, there are *probabilistic mappings* between data sources [24]. Such mappings are likely to increase in the future in large-scale data integration projects, because of the difficulties to compute automatically accurate, deterministic mappings between complex data sources. It is know from the deterministic case that query answering in the presence of LAV ("local as view") mappings is equivalent to query answering using views. Except for some initial results in [21], the query answering problem over probabilistic views is largely unexplored.

Security and Information Leakage The probabilistic data model has been used to reason about information leakage in views, and more generally, in data exchange. Here the data is private and the owner wishes to publish a certain view, and the concern is that the view might leak private information in the data. One approach to measure the amount of information leakage is to model the attacker's background knowledge as a probability space, and to check whether the a posteriori probability of the secret (after seeing the view) is significantly different from the a priori probability: *perfect security* is when the two are equal [43, 42], while *practical security* is when the two are close [18]. The difficulty here is to compute the query's answer when the input probabilities are not even known. A different, yet very important problem, is the design of security policies in the case when the data is uncertain. Today's common practice in defining access control rules is to specify them in terms of certain credentials offered by a user. For example, a rule for preserving private information in RFID data might say "if a user `$U` was at the same location as a user `X` at the same time `T`, and queries the location of `$X` at time `T` then he/she should be granted access to that location". An open problem is to define the right semantics for such access control policies when the credential is probabilistic: if the system has only 70% confidence that the user `$U` was a certain location at time `T`, should it answer the query or should it deny it ?

3. REFERENCES

[1] L. A. Adamic and E. Adar. How to search a social network. *Social Networks*, 27(3):187–203, 2005.

[2] E. Adar and C.Re. Managing uncertainty in social networks. *IEEE Data Engineering Bulletin*, 30(2):15–22, 2007.

[3] R. Ananthakrishna, S. Chaudhuri, and V. Ganti. Eliminating fuzzy duplicates in data warehouses. In *VLDB*, 2002.

[4] L. Antova, C. Koch, and D. Olteanu. MayBMS: Managing incomplete information with probabilistic world-set decompositions In *ICDE*, 2007.

[5] A. Arasu, V. Ganti, and R. Kaushik. Efficient exact set-similarity joins. In *VLDB*, 2006.

[6] D. Barbara, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *IEEE TKDE*, 4(5):487–502, 1992.

[7] O. Benjelloun, A. Das Sarma, A. Halevy, and J. Widom. ULDBs: Databases with uncertainty and lineage. In *VLDB*, 2006.

[8] O. Benjelloun, A. Das Sarma, C. Hayworth, and J. Widom. An introduction to ULDBs and the Trio system. *IEEE Data Eng. Bull*, 29(1):5–16, 2006.

[9] J. Boulos, N .Dalvi, B. Mandhani, S. Mathur, C. Re, and D. Suciu. Mystiq: A system for finding more answers by using probabilities. In *SIGMOD*, 2005.

[10] R. Cavallo and M. Pittarelli. The theory of probabilistic databases. In *of VLDB*, 1987.

[11] S. Chaudhuri, K. Ganjam, V. Ganti, and R. Motwani. Robust and efficient fuzzy match for online data cleaning. In *ACM SIGMOD*, San Diego, CA, 2003.

[12] R. Cheng and S. Prabhakar. Managing uncertainty in sensor databases. *SIGMOD Record*, 32(4), 2003.

[13] T. Choudhury, M. Philipose, D. Wyatt, and J. Lester. Towards activity databases: Using sensors and statistical models to summarize people's lives. *IEEE Data Eng. Bull.*, 29(1):49–58, 2006.

[14] W. Cohen, P. Ravikumar, and S. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, 2003.

[15] Garage Band Corp. `www.ilike.com`.

[16] R. Cowell, P. Dawid, S. Lauritzen, and D. Spiegelhalter, editors. *Probabilistic Networks and Expert Systems*. Springer, 1999.

[17] D. M. Boyd. Friendster and publicly articulated social networking. In *CHI 2004*, 2004.

[18] N. Dalvi, G. Miklau, and D. Suciu. Asymptotic conditional probabilities for conjunctive queries. In *ICDT*, 2005.

[19] N. Dalvi, Chris Re, and D. Suciu. Query evaluation on probabilistic databases. *IEEE Data Engineering Bulletin*, 29(1):25–31, 2006.

[20] N. Dalvi and D. Suciu. Efficient query evaluation on probabilistic databases. In *VLDB*, 2004.

[21] N. Dalvi and D. Suciu. Answering queries from statistics and probabilistic views. In *VLDB*, 2005.

[22] N. Dalvi and D. Suciu. The dichotomy of conjunctive queries on probabilistic structures. In *PODS*, 2007.

[23] N. Dalvi and D. Suciu. Management of probabilistic data: Foundations and challenges. In *PODS*, 2007. (invited talk).

[24] X. Dong, A. Halevy, and C. Yu. Data integration with uncertainty. In *VLDB*, 2007.

[25] Facebook. `www.facebook.com`.

[26] I. Felligi and A. Sunter. A theory for record linkage. *Journal of the American Statistical Society*, 64:1183–1210, 1969.

[27] L. C. Freeman. *The Development of Social Network Analysis: A Study in the Sociology of Science*. Empirical Press, 2004.

[28] N. Fuhr and T. Roelleke. A probabilistic relational algebra for the integration of information retrieval and database systems. *ACM Trans. Inf. Syst.*, 15(1):32–66, 1997.

[29] H. Galhardas, D. Florescu, D. Shasha, E. Simon, and C.A. Saita. Declarative data cleaning: Language, model, and algorithms. In *VLDB*, 2001.

[30] L. Garton, C. Haythornthwaite, and B. Wellman. Studying online social networks. *Journal of Computer-Mediated Communication*, 3, 1997.

[31] L. Getoor. An introduction to probabilistic graphical models for relational data. *IEEE Data Engineering Bulletin, Special Issue on Probabilistic Data Management*, 29(1):32–40, March 2006.

[32] E. Grädel, Y. Gurevich, and C. Hirsch. The complexity of query reliability. In *PODS*, 1998.

[33] T.J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, 2007.

[34] L. Gu, R. Baxter, D. Vickers, and C. Rainsford. Record linkage: Current practice and future directions. In *CMIS Technical Report No. 03/83*, 2003.

[35] R. Gupta and S. Sarawagi. Creating probabilistic databases from information extraction models. In *VLDB*, 2006.

[36] M. Hernandez and S. Stolfo. The merge/purge problem for large databases. In *SIGMOD*, 1995.

[37] B. Hogan, J. A. Carrasco, and B. Wellman. Visualizing Personal Networks: Working with Participant-aided Sociograms. *Field Methods*, 19(2):116–144, 2007.

[38] T. Imielinski and W. Lipski. Incomplete information in relational databases. *Journal of the ACM*, 31:761–791, October 1984.

[39] Linked In. `www.linkedin.com`.

[40] D. Koller. Representation, reasoning, learning. Computers and Thought 2001 Award talk.

[41] L. Lakshmanan, N. Leone, R. Ross, and V.S. Subrahmanian. Probview: A flexible probabilistic database system. *ACM TODS*, 22(3), 1997.

[42] A. Machanavajjhala and J. Gehrke. On the efficiency of checking perfect privacy. In *PODS*, 2006.

[43] G. Miklau, D. Suciu. A formal analysis of information disclosure in data exchange. In *SIGMOD*, 2004.

[44] MySpace. `www.myspace.com`.

[45] Visible Path. `www.visiblepath.com`.

[46] Judea Pearl. *Probabilistic reasoning in intelligent systems*. Morgan Kaufmann, 1988.

[47] S. Singh R. Cheng and S. Prabhakar. U-DBMS: A database system for managing constantly-evolving data. In *VLDB*, 2005.

[48] C. Re, N. Dalvi, and D. Suciu. Efficient Top-k query evaluation on probabilistic data. In *ICDE*, 2007.

[49] C. Re and D.Suciu. Efficient evaluation of having queries on a probabilistic database. In *DBPL*, 2007.

[50] C. Re and D.Suciu. Materialized views in probabilistic databases for information exchange and query optimization. In *VLDB*, 2007.

[51] Sunita Sarawagi. Automation in information extraction and data integration. Tutorial presented at VLDB'2002.

[52] P Sen and A. Deshpande. Representing and querying correlated tuples in probabilistic databases. In *ICDE*, 2007.

[53] L. Valiant. The complexity of enumeration and reliability problems. *SIAM J. Comput.*, 8:410–421, 1979.

[54] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[55] E. et al. Welbourne. Challenges for pervasive RFID-based infrastructures. In *PERTEC Workshop*, March 2007.

[56] B. Wellman. Challenges in Collecting Personal Network Data: The Nature of Personal Network Analysis. *Field Methods*, 19(2):111–115, 2007.

[57] S. White and P. Smyth. Algorithms for estimating relative importance in networks. In *KDD 2003*, 2003.

[58] J. Widom. Trio: A system for integrated management of data, accuracy, and lineage. In *CIDR*, 2005.

[59] William Winkler. The state of record linkage and current research problems. Technical report, Statistical Research Division, U.S. Bureau of the Census, 1999.

[60] E. Zimanyi. Query evaluation in probabilistic databases. *TCS*, 171, 1997.