Sources of Uncertainty in Information Systems *

Amihai Motro
Department of Information and Software Systems Engineering
George Mason University
Fairfax, VA 22030-4444

September 1992

Abstract

We define a simple and general model of information systems, and use this model to classify the various kinds of uncertainty in information systems into two major categories: uncertainty in *descriptions* of the real world and uncertainty in *manipulations* of descriptions. We survey briefly several approaches that have been developed for dealing with these kinds of uncertainty, and we speculate on the reasons that commercial information systems have been slow to incorporate such uncertainty capabilities. Finally, we suggest that relevant research in the area of artificial intelligence may be applicable to information systems, after it is adapted to address the specific concerns and constraints of information systems. An appendix identifies additional instances of uncertainty in information systems.

*This work was supported in part by NSF Grant No. IRI-9007106 and by ARPA grant, administered by the Office of Naval Research under Grant No. N0014-92-J-4038.

1 Introduction

Uncertainty permeates our understanding of the real world. The purpose of information systems is to model the real world. Hence, information systems must be able to deal with uncertainty.

And, indeed, many information systems include capabilities for dealing with some kinds of uncertainty. For example, database systems can represent missing values, information retrieval systems can match information to requests using a "weak" matching algorithm, and expert systems can represent rules that are known to be true only "most" or "some" of the time.

Yet, these capabilities are weak, when compared to the variety and the degree of uncertainty that is encountered in practice. By and large, commercial information systems (e.g., database systems, information retrieval systems, expert systems) have been slow to incorporate capabilities for dealing with uncertainty.

The research community has shown persistent interest in this subject. Yet, research has focused mostly on problems arising in individual types of information systems and their own peculiar kinds of uncertainty. Often, the commonalities across different systems has been ignored, and limited effort has been directed at understanding uncertainty at a more general level.

In this paper we provide a brief survey of uncertainty in information systems. Our goal is to understand and classify the different sources of uncertainty, rather than to provide an extensive survey of all the work that has been done in this area. Moreover, we limit our attention to uncertainty in the context of information systems, not computer systems in general.

Later in this section we provide working definitions of *information system* and *uncertainty*, and we classify uncertainty into several major categories. In Sections 2, 3 and 4 we review several different kinds of uncertainty in each of the categories, and we discuss briefly a variety of solutions that have been advanced. Additional instances of uncertainty in information systems are identified in the Appendix.

Information systems have their distinctive pragmatic constraints, which any approach to uncertainty must observe. In Section 5 we discuss these constraints, and we speculate on the reasons that commercial information systems have been slow to incorporate capabilities for dealing with uncertainty. Understanding these reasons could suggest how to build better systems. Finally, we suggest that work done on the modeling of uncertainty as part of research on artificial intelligence (AI) can be adapted to the needs of information systems (IS).

1.1 Basic Terminology

Information System. An information system is a computer model of some portion of the real world. Like any other model, it abstracts reality to a level warranted by the expected applications. Usually, such models include a declarative component for *describing* the real world, and an operational component for *manipulating* this description. Typical manipulations are (1) *modifications* of the description, either to refine the model or to track any changes that may have occurred in the real world; and (2) *transformations* of the description, to derive implied descriptions.

Thus, an information system can be abstracted as a description D of the real world, a stream of modifications, and a stream of transformations. Each modification m replaces the present description with a new description; each transformation t computes a new description from the present description (without changing the present description).

In this paper, we consider three popular types of information systems: database systems, information retrieval systems, and expert systems. As an example, in a relational database system a description is a set of tables (i.e., a database); a modification can affect either the definition or the contents of tables (i.e., restructuring or update); and a transformation reduces the set of tables into a single table (i.e., query evaluation).

Uncertainty. We assume that uncertainty permeates our *models* of the real world, not the real world itself. We use the term uncertainty to refer to any aspect of the model that cannot be asserted with complete confidence. In particular, uncertainty can be expected (1) in the descriptions of the real world, and (2) in the modifications and transformations of these descriptions. Referring again to the example of relational database systems, these correspond to the data, the restructuring and update transactions, and the queries.

The term uncertainty is intended to generalize many individual concepts that have been defined previously in specific contexts, including imprecision, incompleteness, ambiguity, vagueness, and fuzziness (some of these concepts will be discussed in later sections).

The final objective of any information system is to provide its users with the information they need. In our terminology, such information is always the result $t(D)$ of transforming a description D with a transformation t. Thus, it is the quality of $t(D)$, rather than the quality of D, that should concern the designers and users of information systems. A result $t(D)$ may be uncertain, either because D is uncertain or because t is uncertain. In turn, the uncertainty of D may owe to uncertainty either in the initial description, or in some later modification m.

In the next three sections we discuss uncertainty in these three categories: descriptions, transformations, and modifications.

2 Uncertainty in Descriptions: Classification

Undoubtedly, uncertainty that permeates descriptions has received the most attention. In this section we examine three basic issues regarding description uncertainty; namely, the different *elements* of descriptions that could be affected with uncertainty, the different *sources* for uncertainty, and the different *degrees* of uncertainty. Roughly, we ask *what* is uncertain, *why* is it uncertain, and *how* uncertain is it?

Elements affected by uncertainty. Depending on the model used, descriptions may take different forms, and uncertainty could affect each of them.

Consider, for example, relational databases. The structures of the relational model admit different kinds of uncertainty. The first kind involves uncertainty at the level of data values; for example, the values of SALARY in the relation EARN (EMPLOYEE, SALARY) might be uncertain. The second kind involves uncertainty at the level of the tuple; for example, the values of each of the attributes of the relation ASSIGN (EMPLOYEE, PROJECT) may be certain, but there might be uncertainty about the precise assignment of employees to projects. A third kind involves uncertainty at the level of the relation (the structure); for example, there might be uncertainty whether employees may belong to more than one department, and hence what should be the proper description of this relationship [4].

As another example, consider an information retrieval system that models each document with an identifier and a vector of keywords. There might be uncertainty at the level of a keyword; i.e., the appropriateness of a specific keyword to a given document may be questionable. In addition, there might be uncertainty at the level of an entire document; i.e., the existence of some documents may be in doubt.

As a third example, consider an expert system that models real world knowledge with facts and rules expressed in logic. There might be uncertainty about specific facts (similar to the tuple uncertainty in relational databases), and there might be uncertainty about specific rules; i.e., a rule might be only an approximation of the behavior of the real world.

Sources of Uncertainty. Having excluded the possibility that reality itself is subject to uncertainty, we can always assume that a "perfect" description of any real world object always exists. Thus, uncertain descriptions are solely due to the *unavailability* of these "perfect" descriptions. For example, the precise salary of Tom might be unknown, or the true relationship between Bordeaux wine and good health might be unclear. Yet, within this "generic unavailability", we observe several specific sources of uncertainty [17].

Uncertainty might result from using *unreliable information sources*, such as faulty reading instruments, or input forms that have been filled-out incorrectly (intentionally or inadvertently). In other cases, uncertainty is a result of *system errors*, including input errors,

transmission "noise", delays in processing update transactions, imperfections of the system software, and corrupted data owing to failure or sabotage.

At times, uncertainty is the unavoidable result of information gathering methods that require *estimation* or *judgement*. Examples include the determination of the subject of a document, the digital representation of a continuous phenomenon, and the representation of a phenomenon which is constantly varying.

In other cases, uncertainty is the result of restrictions imposed by the *model*. For example, if the database schema permits storing at most two occupations per employee, descriptions of occupation would exhibit uncertainty. Similarly, the sheer *volume* of the information that is necessary to describe a real world object might force the modeler to turn to approximation and sampling techniques.

Degrees of uncertainty. The relevant information that is available in the absence of certain information may take different forms, each exhibiting a different level of uncertainty. The following discussion is independent of the particular structure affected by uncertainty. It assumes an element e of the description models an object o of the real world. e might be a value, a fact, a tuple, a rule, etc.

Uncertainty is highest when the mere *existence* of some real world object is in doubt. The simplest solution is to ignore such objects altogether. This solution, however, is unacceptable if the model claims to be *closed world* (i.e., objects not modeled do not exist).

Uncertainty is reduced somewhat when each element is assigned a value in a prescribed range, to indicate the certainty that the modeled object exists. When the element is a fact, this value can be interpreted as the *confidence* that the fact holds; when it is a rule, this value can be interpreted as the *strength* of the rule (percent of cases where the rule applies).

Assume now that existence is assured, but some of or all the information with which the model describes an object is *unknown*. Such information has also been referred to as *incomplete*, *missing* or *unavailable*.

Uncertainty is reduced when the information that describes an object is known to come from a limited set of alternatives (possibly a range of values). This uncertainty is referred to as *disjunctive* information. Note that when the set of alternatives is simply the entire "universe", this case reverts to the previous (less informative) case.

Uncertainty is reduced even further when each alternative is accompanied by a number describing the probability that it is indeed the true description (and the sum of these numbers for the entire set is 1). In this case, the uncertain information is *probabilistic*. Again, when the probabilities are unavailable, probabilistic information becomes disjunctive information.

Occasionally, the information available to describe an object is *descriptive* rather than quantitative. Such information is often referred to as *fuzzy* or *vague* information.

3 Uncertainty in Descriptions: Solutions

Space considerations forbid discussion of all the different solutions that have been attempted for accommodating uncertainty in descriptions. We sketch here five approaches that are different from each other; they also exhibit sufficient generality to be applicable in different information systems.

3.1 Null Values

Most information models insist that similar real world objects be modeled with similar descriptions. The simplest example of this approach are models that use tabular descriptions. Each such table models a set of similar real world objects: each row describes a different object, and the columns provide the different components of the description. Often, some elements of a particular description cannot be stated with certainty. Occasionally, this problem may be evaded simply by not modeling any object whose description is incomplete. Often, however, the consequences of this approach are unacceptable, and incomplete descriptions must be admitted. Not surprisingly, incompleteness is a lesser issue in models that do not rely on mandatory information.

The least ambitious approach to admitting incomplete descriptions is to ignore all *partial* information about the uncertain parts of a description that may be available, and to model them with a pseudo-description, called *null*, that denotes uncertainty [18, 15, 37].

Once null values are admitted into descriptions, the model must define the behavior of transformations and modifications in the presence of nulls. This is not always a simple task. For example, an extension to the relational calculus that is founded on a new three-valued logic [7], has been the subject of criticism [8]. Inference in incomplete databases is discussed in [10]. Updates of incomplete databases are discussed in [1].

Various refinements of this approach have been suggested. For example, a distinction has been made between incompleteness owing to *unavailability* and incompleteness owing to *nonexistence*. Similarly, it has been suggested to use *distinct* nulls for unavailable descriptions that are known to be identical. These and other refinements constitute attempts to apply whatever partial information that is available.

3.2 Certainty Factors

Certainty factors denote confidence in various elements of the description. They offer a simple tool for representing uncertainty, and have thus been applied in both information retrieval systems and in expert systems.

In information retrieval systems, certainty factors (often called *weights*) have been used to denote confidence that a specific keyword describes a given document (or alternatively, to denote the strength with which this keyword applies to the document) [30, 34]. Methods have even been developed for computing certainty factors automatically, by scanning documents and applying keyword counting techniques. The manipulation of these certainty factors is relatively simple, as they are easily accommodated in the vector space models that are often used in information retrieval (see Section 3.4 below).

In expert systems, certainty factors have been used to denote confidence that stated facts and rules indeed describe real world objects [12]. Such factors are usually declared by the knowledge engineers as part of the knowledge acquisition process, but can also be derived automatically as part of a knowledge discovery process [24]. The manipulation of certainty factors in expert systems is often straightforward; for example, assuming certainty factors in the range [0,1], when a rule with certainty p is applied to a fact with certainty q, the generated fact is assigned a certainty factor $p \cdot q$. Pragmatic considerations may have been the reason that commercial expert systems usually prefer this mostly informal representation of uncertainty, over more formal approaches that are based on probability theory.

3.3 Possibility and Probability

Fuzzy set theory offers a comprehensive approach to modeling uncertainty in information systems. The approach described here is derived from [26, 36, 27].

The basic concept of fuzzy set theory is the *fuzzy set*. A fuzzy set F is a set of elements, where each element has an associated value in the interval [0,1] that denotes the *grade* of its membership in the set. For example, a fuzzy set may include the elements 20, 30, 40, and 50, with grades of membership 1.0, 0.7, 0.5 and 0.2, respectively.

As a relation is a subset of the product of several domains, one approach is to define relations that are fuzzy subsets of the product of fuzzy domains. Since each such relation is a fuzzy set, each of its tuples is associated with a membership grade. This definition admits uncertainty at the tuple level. For example, the tuple (Dick, Pascal) belongs to the relation PROFICIENCY(PROGRAMMER, LANGUAGE) with membership grade 0.9 (alternatively, this tuple may be interpreted as stating that Dick's proficiency in Pascal is 0.9).

Consider the fuzzy set defined earlier, and assume it is named YOUNG. It is also possible to interpret this set as the definition of the term "young": it is a term that refers to 20 year olds with possibility 1.0, to 30 year olds with possibility 0.7, etc. Thus, fuzzy sets may be applied to describe imprecise terms.

Consider now standard (nonfuzzy) relations, but assume that the elements of the domains are not values, but fuzzy sets of values. This definition admits uncertainty at the data value level. Having fuzzy sets for values permits specific cases where a value is one of five kinds: (1) A set; for example, the value of DEPARTMENT can be {shipping, receiving} or the value of

SALARY can be 40,000–50,000. Note that the interpretation of such sets is purely *disjunctive*: exactly one element of the set is the correct value. (2) A fuzzy value; for example, the value of AGE can be young. (3) An estimate; for example, the value of SMART can be 0.8. (4) A null value. (5) A simple value.

Finally, by defining a fuzzy relation as a fuzzy subset of the product of fuzzy domains of fuzzy sets, both kinds of uncertainty can be accommodated.

To manipulate fuzzy databases, the standard relational algebra operators must be extended to fuzzy relations. The first approach, where relations are fuzzy sets but elements of domains are "crisp", requires simple extensions to these operators. The second approach, where relations are crisp but elements of domains are fuzzy, introduces more complexity because the "softness" of the values in the tuples creates problems of value identification (e.g., in the join, or in the removal of replications after projections). Also, in analogy with standard mathematical comparators such as $=$ or $<$, which are defined via sets of pairs, the second approach introduces fuzzy comparators such as *similar-to* or *much-greater-than*, which are defined via fuzzy sets of pairs. These fuzzy operators offer the capability of expressing fuzzy (uncertain) retrieval requests.

In [2] a model is described that handles uncertainty with traditional probability distributions (rather than the possibility distributions of the fuzzy models). A *probability distribution function* of a variable X over a domain D assigns each value $d \in D$ a value between 0 and 1, as the probability that $X = d$. One important difference between probability and possibility distribution functions is that the sum of the probabilities assigned to the elements of X must be exactly 1. The definition of a probabilistic database is similar to the second definition of fuzzy databases: standard relations, but with domain values that are, in general, probability distribution functions. A feature of this model is that it allows probability distributions that are incompletely specified: each such distribution is complemented with a *missing value* which is assigned the balance of the probability.

3.4 Distances

An approach to uncertainty that has been applied successfully to both databases systems and information retrieval systems handles uncertainty with *distance*. The basic idea is to model the real world with apparently-certain descriptions, and to rely on definitions of distances among descriptions to create *neighborhoods* of descriptions.

Thus, any uncertainty about a real world object o is ignored, and an apparently-certain description of it e is stored. It is then hoped that this "negligence" would be compensated by having e somewhere in the neighborhood of the true description. When a request for information specifies this true description, e would be retrieved, along with the other neighbors of the true description.

As an example, consider an information retrieval system that describes documents with sets of keywords [30, 34]. Such systems often represent keyword sets with vectors: the dimension of each vector is the number of possible keywords, and a specific vector position is 1 if a particular keyword is in the set and 0 otherwise. Often, there is uncertainty whether a specific vector is the true description of a given document. By establishing a distance among document descriptions, usually with some vector metric, and retrieving all the information in the neighborhood of a request-vector, the negative effects of inaccuracies in the description are diminished.

As another example, consider relational database systems. Such systems describe objects with tuples, and often there is uncertainty regarding the value of some attribute in a given tuple. It is possible to establish a distance among the elements of the domain of this attribute. Then, when a query specifies a value of this attribute, all the tuples would be retrieved, whose value for that attribute is in the neighborhood of the specified value [23].

We assumed here that descriptions are subject to uncertainty (which is ignored) and requests are certain. The same solution applies when descriptions are certain and requests are subject to uncertainty. Such requests would be specified with apparent-certainty, and would be answered with the neighborhood of the request. Again, the negative effects of inaccuracies in the request would be moderated. Uncertainty in requests is discussed in more detail in Section 4.1.

A refinement of this general method is to admit certainty factors (Section 3.2) in the descriptions and in the requests. For example, vectors describing keyword sets use values in the range [0,1] to denote the certainty that a specific keyword applies to the given document (or, alternatively, to denote the strength with which this keyword applies). Similarly, request-vectors use such values to denote the weights of various keywords in the overall request.

3.5 Soundness and Completeness

The final approach we discuss has been developed in the context of relational databases [21]. Instead of modeling "imperfections" in the information (i.e., uncertainty), it suggests declaring the portions of the database that are prefect models of the real world (and thereby the portions that are possibly imperfect).

Thus, like distances (Section 3.4) and unlike incompleteness (Section 3.1), certainty factors (Section 3.2) or fuzziness (Section 3.3), the descriptions themselves have no special features for uncertainty; i.e., they appear certain. However, meta-information provides the distinction between certain and uncertain information. (Recall that fuzziness and distances also require meta-information: the definitions of fuzzy terms or the measures of proximity.)

With this information included in the database, the database system can *qualify* the accuracy of the answers it issues in response to queries: each answer is accompanied by statements that define the portions that are guaranteed to be perfect.

This approach interprets certainty, which it terms *integrity*, as a combination of *soundness* and *completeness*. A description is sound, if it includes *only* information that occurs in the real world; a description is complete, if it includes *all* the information that occurs in the real world. Hence, a description has integrity, if it includes the whole truth (completeness) and nothing but the truth (soundness).

The approach uses the mechanism of *views*, to specify the portions of the database or the portions of an answer that have integrity. It describes a technique for inferring the views of individual answers that are guaranteed to have integrity, from the views of the entire database that are known to have integrity.

The concept of view completeness is similar to an assumption that a certain view of the database is *closed world*; the notion of view soundness is shown to be a generalization of standard database integrity constraints.

4 Uncertainty in Manipulations

4.1 Transformations

Transformations are operations that derive new descriptions from stored descriptions. The most frequent type of transformation are requests from users for information. Uncertainty of requests may have different sources.

Insufficient knowledge of the system. At times, users of information systems have insufficient knowledge of the information system they are using: they might not have a clear idea of the information available in the system (or how it is organized), or they might not know how to formulate their requests with the tools provided by the system.

Requests for information formulated by such *naive* users exhibit a high level of uncertainty. They range from requests that cannot be interpreted by the system (for reasons that are either syntactical or semantical) to requests that do not achieve correctly the intentions of their authors (or achieve them only in part).

Vague specifications. Regardless of their level of expertise, occasionally users may try to access an information system with only a *vague* idea of the information they seek. For example, a user may be accessing an electronic catalogue for a product that would "interesting" or "exceptional". Alternatively, users could have a clear idea of the information they want, but might lack the information necessary to specify it to the system. An example is a user who wishes to look up the meaning of a word in a dictionary, but cannot provide its correct spelling.

To summarize, we distinguish among (1) uncertainty of the *information available* (or how it is organized), (2) uncertainty of the *information needed* (or how to specify it), and (3) uncertainty of the *formalisms*.

To address all these, the approach has been to develop alternative access tools. Browsers allow users to access information in either situation discussed above [19, 28, 9]. Interactive query constructors conduct user-system dialogues to define the request satisfactorily [35]. Vague query processors allow users to embed uncertain specifications in their request; e.g., neighborhood queries (see Section 3.4 above) [14, 23]. Error-tolerant interfaces use relaxed formalisms in their interpretation of requests [20].

Answer Uncertainty. As mentioned above, even after a request for information had been accepted by the system and its answer delivered to the user, uncertainty might still persist, because it is not always possible to verify that the request indeed achieved correctly the intention of the user. Often, the only assurance that the information delivered is the information needed, is that the user is somewhat familiar with it. In the absence of such familiarity, uncertainty will persist.

Hence, one must accept that there would be frequent instances where the answer that is delivered is inaccurate, yet both the system and the user are unaware of this uncertainty. Often, the uncertainty of apparently-certain answers is revealed only when a conflicting answer to the same request is received from another information system.

Uncertain answers can also be the result of the methods used by the system to process requests. For example, an information system might allocate only limited computational resources to process a request [16], or processing might apply randomizations, sampling or other estimation techniques. In each case, the answers would exhibit uncertainty.

Finally, it is sometime considered advantageous to sacrifice accuracy for the sake of *simplicity*. Recent research on *intensional answers* [6, 25, 22] has focused on the generation of abstract answers that describe the exhaustive answers compactly, albeit imperfectly.

4.2 Modifications

Modifications are operations that affect the descriptions stored in information systems, either to refine them or to update them. Like transformations (requests for information), modifications are defined by users. Unlike transformations, they are usually experts.

Thus, insufficient knowledge of the system is rarely a source for uncertain modification requests. Vaguely specificed modifications, however, are entirely possible. One example is a request to add information that is uncertain; e.g., "the new manager is either Paul or John". Another example is a request to delete information, with uncertainty about what exactly should be deleted; e.g., "some of the telephone numbers are no longer valid".

However, this kind of uncertainty is not any different from the description uncertainty, that was discussed in Sections 3 and 4. Thus, the first request would be accommodated as any information of the kind "exactly one of the following values holds", and the second request would be accommodated as any information of the kind "some of the following values hold". (Of course, if the system cannot model these kinds of uncertainty, then it would not be able to handle these modifications.)

5 Conclusion

5.1 Hindrances

Commercial information systems have been slow to incorporate uncertainty capabilities. While solutions based on fuzzy set theory (Section 3.3) are gaining acceptance in several technologies, commercial information systems have not embraced this solution yet. In database systems, the only capabilities widely available are for handling null values (Section 3.1), and for specifying imprecise queries with regular expressions. Most commercial information retrieval systems do not have the uncertainty capabilities that come with the definition of distances (Section 3.4). The situation is somewhat better in expert systems, where commercial systems often permit certainty factors in their facts and rules (Section 3.2).

Examining the possible reasons for this slow acceptance may suggest directions for further research. First, IS practitioners are concerned primarily with the *performance* of their systems. However, many of the algorithms for matching uncertain data or for processing uncertain requests are extremely complex and inefficient.

Practitioners are also concerned with *compatibility*. This dictates that capabilities for accommodating uncertainty should be offered as strict extensions of existing standards.

Third, database practitioners have often been dissatisfied with various *idiosyncratic implementations* of uncertainty capabilities (minor examples are the inability to specify an incomplete date value, or the inability to sort answers with null values flexibly). This may have had a chilling effect on further implementations.

Another hindrance for database systems with uncertainty capabilities may lie in the *expectations of users*. A fundamental principle of database systems has been that queries and answers are never open to "subjective" interpretations, and users of database systems have come to expect their queries to be interpreted unambiguously and answered with complete accuracy. In contrast, users of information retrieval systems would be pleased with a system that delivers a high rate of recall (proportion of relevant material retrieved) and precision (proportion of retrieved material that is relevant). Similarly, users of expert systems are aware that conclusions offered by "automated experts" are often questionable, and would be satisfied with systems that have shown to have a high rate of success. A database system that

must adhere to the principles of unambiguity and complete accuracy cannot accommodate the full range of uncertain information; for example, it can accommodate null values or disjunctive data, but not (because of its more subjective nature) probabilistic or fuzzy data.

5.2 Information Systems and Artificial Intelligence

Modeling uncertainty in information systems involves a classical dilemma. On one hand, we want a model as rich and as powerful as possible; for example, a single concept of information that is capable of representing elegantly all known kinds of uncertain information, as well as certain information. On the other hand, information systems must abide by crucial constraints of simplicity and efficiency, both of their descriptions and of the manipulations of their descriptions.

Like information systems, the field of artificial intelligence too has been concerned with modeling our knowledge of the real world. Numerous theories have been developed, mostly they are founded on non-classical logics or on probability theory [3, 31].

It might be said that, traditionally, research in artificial intelligence has been emphasizing rich and powerful models, striving to model accurately all the minute nuances of reality. On the other hand, research in information systems has been emphasizing economical representations with lower expressivity, but with small representational overhead and high processing efficiency.

A notable case in point are the so-called *semantic data models* that were defined in the late 70's to enhance the modeling capabilities of early database systems [11, 13]. These models incorporated modeling features, such as generalization and aggregation hierarchies, that had been adapted from research in artificial intelligence. An even earlier example is the adaptation of *semantic networks* for databases [29]. A third and more recent example are *knowledge-rich database systems*, that incorporate inference rules expressed in mathematical logic [32, 33, 5]. Not surprisingly, a major research thrust among database researchers that have been working in this area, has been the development of *highly efficient* inference techniques for a *limited* class of rules.

Thus, information systems may be said to have embraced "poor-person's AI", or, more graciously, to have adapted AI concepts to work under the strict constraints of information systems. It is my conjecture that the management of uncertainty is another instance where information systems may benefit from theories developed for artificial intelligence.

Appendix: Additional Issues of Uncertainty

This appendix identifies various issues of uncertainty found in information systems. It should serve to expand and illustrate the discussion of uncertainty in this paper. In some cases, the issues raised have been treated in research works. In other cases, the issues are being investigated, or are awaiting research.

The process of *database schema design* might involve uncertainty (see Section 2); for instance, a designer may not know what kinds of addresses people will have, though it may be assumed that the majority will be American. This problem could be handled by schema design languages with mechanisms for exceptions. An important constraint is that handling information with exceptions should not require more time.

Description uncertainty could be due to *incompleteness* of information (Section 2); for instance, it might be known only that Joe is married to either Mary or Sue and has 2 or 3 children. The answer to a query such as "married persons with at most 4 children" must then include Joe. This situation may be handled in description logics, such as those offered in Classics. Query evaluation would involve subsumption.

Abstract answers sacrifice precision for clarity (Section 4.1). Consider the query "who can teach CS410?". One approach is to create a hypothetical individual who can teach this course (as part of the query), then deduce facts about this individual, and return these facts as part of the answer.

Databases with *incomplete information* may employ *plausible reasoning* to determine the missing information. In this method, rules that have been declared by experts or discovered in the database (Section 3.2) are used to fill-in some of the nulls; for example, the status of students could be inferred from their enrollments, and the salary of employees could be inferred from combinations of age, position and qualifications. Of course, the derived information would still exhibit uncertainty (which would be quantified by the reasoning process), but nevertheless the situation would have progressed from total uncertainty (null values) to quantified uncertainty (certainty factors).

The use of unreliable sources of information (Section 2) raises issues of *precedence*: should a present description be updated by a new description, that has been obtained from a less reliable source? The management of uncertain databases may benefit from the establishment of *certainty constraints*, that would establish precedence and thresholds, and, in general, control the degree of uncertainty that would be tolerated.

Probabilistic data models can be used to represent certain kinds of uncertain information (Section 3.3). Uncertainty resulting from *conflicting* information sources (See Section 4.1) can then be translated into a method for *combining* probabilistic relations.

The information stored in databases includes also *integrity constraints*, for stating the correctness criteria of applications; for instance, a constraint may require that at all times

the value X should be identical to the value Y. When X and Y are located at different sites, practical considerations (i.e., site unavailability) suggest that these constraints might be relaxed in a limited way. Such *approximate constraints* introduce uncertainty, as now a value at the disconnected site is not known exactly. It has been suggested that approximate constraints improve performance and information availability in distributed environments.

In scientific and statistical databases, uncertainty can often be treated as *mathematical inaccuracy*, and distance metrics can be defined that measure (or estimate) the divergence of the description from the real world. Assuring maximal accuracy for every transaction could be costly; on the other hand, different kinds of applications may tolerate different levels of inaccuracy. Consequently, it may be beneficial to allow requests to specify the *level of accuracy* that must be maintained. Possibly, requests that exceed the present level of accuracy will force the database to update its description.

Such treatment of inaccuracy also raises traditional numerical issues, such as *error propagation* across a database, and the *accumulation of errors* in queries that require complex calculations (Section 4.1).

Data errors and *discretization* have been mentioned as contributors to uncertainty (Section 2). In some applications, their effect can be reduced by *increasing* the amount of data, and by using *statistical methods* for identifying the data with very high uncertainty.

The accuracy of *knowledge* (either declared or discovered) could be increased by using *knowledge constraints*. Similar to *integrity constraints* that protect databases from inconsistent data, knowledge constraints would reject inconsistent knowledge. The risk, of course, is that the constraints themselves would be inaccurate, and would thus reject proper knowledge.

Discovered knowledge about a particular phenomenon might be inaccurate, because variables that are essential to the particular phenomenon are *missing*, or inversely, because variables that are *irrelevant* to the phenomenon are considered. Various statistical methods have been proposed for identifying both kinds of variables. Also, accurate knowledge might elude discovery, if its pertinent variables cannot be *isolated* (i.e., stated independently) from other phenomena.

In human-system communications (Section 4.1) there is often uncertainty about the *subject of discourse*. Human bureaucracies and computer systems tend identify objects with artificial identifiers (such as social security numbers or internal system identifiers). These are guaranteed to be unique, but have little or no meaning to humans, who tend to identify objects with information that is meaningful, concise, and directly related to the object. The problem of identifying objects in this manner may be investigated in the framework of graph theory, where information objects are modeled by labeled nodes, and labeled edges express the relationships among objects. The problem is then to identify a compact subgraph structure around a given node that uniquely identifies it (the issue of meaningfulness still needs to be considered).

Acknowledgements. The Appendix was compiled from contributions by Alex Borgida, Alessandro D'Atri and Laura Tarantino, Robert Demolombe, Hector Garcia-Molina, Gregory Piatetsky-Shapiro, and Doron Rotem.

References

[1] S. Abiteboul and G. Grahne. Update semantics for incomplete databases. In *Proceedings of the Eleventh International Conference on Very Large Data Bases* (Stockholm, Sweden, August 21–23), pages 1–12, Morgan Kaufmann, Los Altos, California, 1985.

[2] D. Barbara, H. Garcia-Molina, and D. Porter. The management of probabilistic data. *IEEE Transactions on Knowledge and Data Engineering*, (to appear in 1992).

[3] Lea Sombe (P. Besnard, M.-O. Cordier, D. Dubois, L. Farinas del Cerro, C. Froidevaux, Y. Moinard, H. Prade, C. Schwind, and P. Siegel). Special issue on reasoning under incomplete information in artificial intelligence. *International Journal of Intelligent Systems*, 5(4), September 1991.

[4] A. Borgida. Language features for flexible handling of exceptions in information systems. *ACM Transactions on Database Systems*, 10(4):565–603, December 1985.

[5] S. Ceri, G. Gottlob, and L. Tanka. *Logic Programming and Databases*. Springer-Verlag, Berlin, Germany, 1990.

[6] L. Cholvy and R. Demolombe. Querying a rule base. In *Proceedings of the First International Conference on Expert Database Systems* (Charleston, South Carolina, April 1–4), pages 365–371, Institute of Information Management, Technology and Policy, University of South Carolina, Columbia, South Carolina, 1986.

[7] E. F. Codd. Extending the database relational model to capture more meaning. *ACM Transactions on Database Systems*, 4(4):397–434, December 1979.

[8] C. J. Date. NOT is not "not"! In *Relational Database Writings 1985–1989*, Addison Wesley, Reading, Massachusetts, 1990.

[9] A. D'Atri, A. Motro, and L. Tarantino. *ViewFinder: an Object Browser*. Technical Report, Department of Information and Software Systems Engineering, George Mason University, June 1992.

[10] R. Demolombe and L. Farinas del Cerro. An algebraic evaluation method for deduction in incomplete data bases. *Journal of Logic Programming*, 5(??):183–205, ?? 1988.

[11] M. Hammer and D. McLeod. Database description with SDM: a semantic database model. *ACM Transactions on Database Systems*, 6(3):351–386, September 1981.

[12] P. Harmon and D. King. *Expert Systems — Artificial Intelligence in Business*. John Wiley & Sons, New York, New York, 1985.

[13] R. Hull and R. King. Semantic database modeling: survey, applications and research issues. *Computing Surveys*, 19(3):201–260, September 1987.

[14] T. Ichikawa and M. Hirakawa. ARES: a relational database with the capability of performing flexible interpretation of queries. *IEEE Transactions on Software Engineering*, SE-12(5):624–634, May 1986.

[15] T. Imielinski. Incomplete information in logical databases. *Data Engineering*, 12(2):29–40, June 1989.

[16] T. Imielinski. Intelligent query answering in rule based systems. *Journal of Logic Programming*, 4(3):229–257, September 1987.

[17] S. Kwan, F. Olken, and D. Rotem. *Uncertain, Incomplete, and Inconsistent Data in Scientific and Statistical Databases*. Technical Report, Lawrence Berkeley Laboratories, Berkeley, California, 1992.

[18] D. Maier. *The Theory of Relational Databases*. Computer Science Press, Rockville, Maryland, 1983.

[19] A. Motro. BAROQUE: a browser for relational databases. *ACM Transactions on Office Information Systems*, 4(2):164–181, April 1986.

[20] A. Motro. FLEX: a tolerant and cooperative user interface to databases. *IEEE Transactions on Knowledge and Data Engineering*, 2(2):231–246, June 1990.

[21] A. Motro. Integrity = validity + completeness. *ACM Transactions on Database Systems*, 14(4):480–502, December 1989.

[22] A. Motro. Intensional answers to database queries. *IEEE Transactions on Knowledge and Data Engineering*, (to appear in 1992).

[23] A. Motro. VAGUE: a user interface to relational databases that permits vague queries. *ACM Transactions on Office Information Systems*, 6(3):187–214, July 1988.

[24] G. Piatetsky-Shapiro. Discovery, analysis and presentation of strong rules. In G. Piatetsky-Shapiro and W. Frawley, editors, *Knowledge Discovery in Databases*, pages 227–248, AAAI Press/MIT Press, Menlo Park, California, 1991.

[25] A. Pirotte, D. Roelants, and E. Zimanyi. Controlled generation of intensional answers. *IEEE Transactions on Knowledge and Data Engineering*, 3(2):221–236, June 1991.

[26] H. Prade and C. Testemale. Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries. *Information Sciences*, 34(2):115–143, November 1984.

[27] K. V. S. V. N. Raju and A. Majumdar. Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems. *ACM Transactions on Database Systems*, 13(2):129–166, June 1988.

[28] T. R. Rogers and R. G. G. Cattell. *Object-Oriented Database User Interfaces*. Technical Report, Information Management Group, Sun Microsystems, 1987.

[29] N. Roussopoulos and J. Mylopoulos. Using semantic networks for data base management. In *Proceedings of the First International Conference on Very Large Data Bases* (Framingham, Massachusetts, September 22–24), pages 144–172, ACM, New York, New York, 1975.

[30] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, New York, 1983.

[31] P. Smets and M. R. B. Clarke (Guest Editors). Special issue on uncertainty, conditionals and non-monotonicity. *Journal of Applied Non-Classical Logics*, 1(2), 1991.

[32] J. D. Ullman. *Database and Knowledge-Base Systems, Volume I*. Computer Science Press, Rockville, Maryland, 1988.

[33] J. D. Ullman. *Database and Knowledge-Base Systems, Volume II*. Computer Science Press, Rockville, Maryland, 1989.

[34] C. J. van Rijsbergen. *Information Retrieval (Second Edition)*. Butterworths, London, 1979.

[35] M. D. Williams. What makes RABBIT run? *International Journal of Man-Machine Studies*, 21(4):333–352, October 1984.

[36] M. Zemankova and A. Kandel. Implementing imprecision in information systems. *Information Sciences*, 37(1,2,3):107–141, December 1985.

[37] R. Zicari. Incomplete information in object-oriented databases. *SIGMOD Record*, 19(3):5–16, September 1990.