The VLDB Journal (2003) / Digital Object Identifier (DOI) 10.1007/s00778-003-0115-z

A framework for modeling and evaluating automatic semantic reconciliation

Avigdor Gal[1], **Ateret Anaby-Tavor**[1], **Alberto Trombetta**[2], **Danilo Montesi**[3]

[1] Technion – Israel Institute of Technology, Technion City Haifa 32000, Israel
(e-mail: avigal@ie.technion.ac.il)
[2] University of Insubria, Via Ravasi, 2, 21100 Varese Italy
[3] University of Camerino, Via del Bastione, 3, 62032 Camerino, Italy

Edited by ♣. Received: ♣ / Accepted: ♣
Published online: ♣♣ 2003 –

Abstract. The introduction of the Semantic Web vision and the shift toward machine understandable Web resources has unearthed the importance of automatic semantic reconciliation. Consequently, new tools for automating the process were proposed. In this work we present a formal model of semantic reconciliation and analyze in a systematic manner the properties of the process outcome, primarily the inherent uncertainty of the matching process and how it reflects on the resulting mappings. An important feature of this research is the identification and analysis of factors that impact the effectiveness of algorithms for automatic semantic reconciliation, leading, it is hoped, to the design of better algorithms by reducing the uncertainty of existing algorithms. Against this background we empirically study the aptitude of two algorithms to correctly match concepts. This research is both timely and practical in light of recent attempts to develop and utilize methods for automatic semantic reconciliation.

Keywords: Semantic interoperability – Ontology versioning – Mapping

1 Introduction and motivation

The ambiguous interpretation of concepts describing the meaning of data in data sources (e.g., database schemata, XML DTDs, RDF schemata, and HTML form tags) is commonly known as *semantic heterogeneity*. Semantic heterogeneity, a well-known obstacle to data source integration [10], is resolved through a process of *semantic reconciliation*, which matches concepts from heterogeneous data sources. Traditionally, semantic reconciliation was performed by a human observer (a designer, a DBA, or a user) [34,59] due to its complexity [10]. However, manual reconciliation (with or without computer-aided tools) tends to be slow and inefficient in dynamic environments and does not scale for obvious reasons. Therefore, the introduction of the Semantic Web vision and the shift toward machine understandable Web resources has unearthed the importance of automatic semantic reconciliation. Consequently, new tools for automating the process, such as Cupid [43], GLUE [15], and OntoBuilder [49], were proposed. In this work we provide a formal model of semantic reconciliation and analyze in a systematic manner the properties of the process, primarily the inherent uncertainty in the process outcome. An important feature of this research is the identification and analysis of factors that impact the effectiveness of algorithms for automatic semantic reconciliation, leading, it is hoped, to the design of better algorithms by reducing the uncertainty of existing ones. Against this background we empirically study the aptitude of two algorithms to correctly match concepts. As will be argued below, this research is both timely and practical in light of recent attempts to develop and utilize methods for automatic semantic reconciliation.

To illustrate our approach, consider the following simplified example, given in terms of the relational model. We shall use this example throughout this article to highlight various aspects of the proposed framework.

Example 1 (Heterogeneous schemata and mappings). Consider two simplified schemata, each consisting of one relation with car rental information from Avis and Alamo.[1]

`AvisRental(RentalNo`: integer, `PickUpLocationCode`: char(20), `PickUpDate`: date, `PickUpHour`: {0, 1, ..., 23}, `PickUpMinutes`: {0, 15, 30, 45}, `ReturnDate`: date, `ReturnHour`: {0, 1, ..., 23}, `ReturnMinutes`: {0, 15, 30, 45}, `Price`: integer(4:2))

`AlamoRental(RentalNo`: integer, `PickUpLocation`: char(20), `PickUp-Date`: date, `PickUpHour`: {0, 1, ..., 23}, `PickUpMinutes`: {0, 10, 20, 30, 40, 50}, `DropoffDate`: date, `DropoffHour`: {0, 1, ..., 23}, `DropoffMinutes`: {0, 10, 20, 30, 40, 50}, `Price`: integer(4:2))

Manual analysis of these two schemata would most likely yield the following schema equivalence constraints, mapping the terms of one schema into the other:

`AvisRental(RentalNo, PickUpLocationCode, PickUpDate, PickUpHour, PickUpMinutes, ReturnDate, ReturnHour, ReturnMinutes, Price)`$\approxeq$ `AlamoRental (RentalNo, PickUpLocation, PickUp-Date, PickUpHour, PickUpMinutes, DropoffDate, DropoffHour, DropoffMinutes, Price)`

[1] The schemata are based on Web forms of the two car rental companies in 2001.

Manual semantic reconciliation overcomes mismatches in attribute names. For example, `ReturnDate` is mapped to `DropoffDate`. Also, differences of domains, e.g., {0, 15, 30, 45} and {0, 10, 20, 30, 40, 50}, do not serve as a barrier to a human observer in identifying identical concepts. In contrast, as discussed in [14,43,49] and others, the outcome of automatic analysis of the two schemata depends on the tools of evaluation. In particular, the matching of `PickUpDate` with `PickUp-Date` can be easily achieved using string matching and an information retrieval technique known as *dehyphenation*, yet the matching of `DropoffDate` and `ReturnDate` may require the assistance of a thesaurus or machine learning techniques to identify Dropoff and Return as synonyms. □

As demonstrated in Example 1, automatic matching may carry with it a degree of uncertainty, as it is based on syntactic, rather than semantic, means. For example, OntoBuilder [30], a tool for ontology matching among Web sites, utilizes, among other techniques, domain matching to recognize "similar" terms in ontologies. The underlying assumption is that when two terms take values from the same domain (say, the integers), they are more likely to be similar than if they take their values from completely different domains. Using such a method, it may be possible to erroneously match noncomparable terms such as height and weight. As another example, consider name matching, a common method in tools such as Cupid [43], OntoBuilder, and Protégé [28]. With name matching, one assumes that similar terms (or attributes) have similar (or even identical) names. However, the occurrence of synonyms (e.g., `remuneration` and `salary` as alternative terms) and homonyms (e.g., `age` referring to either human age or wine age) may trap this method into erroneous matching.

The proposed model, to be given in detail in Sect. 4, utilizes a fuzzy framework to model the uncertainty of the matching process outcome. For example, given two attribute sets $\mathcal{A}$ and $\mathcal{A}'$, we associate a confidence measure, normalized between 0 and 1 with any mapping among attributes of $\mathcal{A}$ and $\mathcal{A}'$. Therefore, given two attributes $A \in \mathcal{A}$ and $A' \in \mathcal{A}'$, we say that we are μ-*confident* in the mapping of A and A' (denoted $A \backsim_{\mu_{att}} A'$) to specify our belief in the mapping quality. We assume that a manual matching is a perfect process, resulting in a *crisp* matching, with $\mu_{att} = 1$.[2] As for automatic matching, a hybrid of algorithms, such as presented in [14,43,49], or adaptation of relevant work in proximity queries (e.g., [3,11]) and query rewriting over mismatched domains (e.g., [12,13]) can determine the level of μ_{att}.

Example 2 (Quantifying imprecision). To illustrate the proposed method for quantifying imprecision, consider a mapping that is based on substring matching. The confidence in the mapping of two attributes A_1 and A_2 is defined symmetrically as the maximum size of a matching substring in A_1 and A_2 divided by the maximum number of characters in either A_1 or A_2. Consider next the schemata in Example 1, and let $A_1 =$ `PickUp-Date` and $A_2 =$ `PickUpDate`. Then, $\mu_{att} = \frac{6 \text{ (for PickUp)}}{1}$1(for `PickUp-Date`) $= 0.55$, due to the hyphen in A_1. However, by applying the dehyphenation technique first, our confidence in the mapping increases dramatically to $\mu_{att} = 1$. □

Identifying a confidence measure μ in and of itself is insufficient for matching purposes. One may claim, and justly so, that the use of syntactic means to identify semantic equivalence may be misleading in that a mapping with a high μ can be less precise, as conceived by an expert, than a mapping with a lower μ. Thus the main contribution of this paper lies in demonstrating through theoretical and empirical analysis that for a certain family of "well-behaved" mappings (termed here *monotonic*), one can safely interpret a high confidence measure as a good semantic mapping. An immediate consequence of this result is the establishment of corroboration for the quality of mapping techniques based on their ability to generate monotonic mappings. We apply this corroboration on two algorithms and report on our experiences in Sect. 6. From our findings we can conclude that matching algorithms that generate monotonic ("well-behaved") mappings are well suited for automatic semantic reconciliation.

The rest of the paper is organized as follows. Section 2 surveys related work, followed by some preliminaries, presenting basic concepts in fuzzy-based applications (Sect. 3). Section 4 introduces the proposed framework for automatic semantic reconciliation. We formally define confidence relations (primitive and compound) as fuzzy relations and demonstrate these concepts by defining confidence relations among data values, domains, individual attributes, and schema mappings. We next define a class of monotonic mappings in Sect. 5 for which we show that fuzzy matching reflects the precision of the mapping itself and analyze some properties of compound confidence relations. In particular, we provide a justification, in retrospect, for the common use of weighted bipartite matching in identifying the best mapping. Section 6 introduces our empirical results, experimenting with two matching algorithms. The paper is concluded in Sect. 7.

2 Background and related work

This study builds upon two existing bodies of research, namely, heterogeneous databases and ontology design, each of which is elaborated below. In addition, we briefly survey some alternatives to the proposed framework.

2.1 Heterogeneous databases

The evolution of organizational computing, from "islands of automation" into enterprise-level systems, has created the need to homogenize heterogeneous databases. More than ever before, companies are seeking integrated data that go well beyond a single organizational unit. In addition, high percentages of the organizational data are supplied by external resources (e.g., the Web and extranets). Data integration is thus becoming increasingly important for decision support in enterprises [8]. The increasing importance of data integration also implies that databases with heterogeneous schemata face

[2] This is, obviously, not always the case. In the absence of sufficient background information, human observers are bound to err as well. However, since our methodology is based on comparing machine-generated mappings with a mapping as conceived by a human expert, and the latter is based on human interpretation, we keep this assumption.

an increasing prospect that their data integration process will not manage semantic differences effectively. This may result, at least to some degree, in mismatching of schema elements. Hence methods for schema matching should take into account a certain level of uncertainty. Current research into heterogeneous databases is, however, largely geared toward deterministic semantic resolution [4,29,36,50,54], which may not effectively scale in computational environments that require rapid response in dynamically changing schemata. In addition, schema descriptions differ significantly among different domains. Accordingly, it is often said that the next main challenge in the semantic matching arena is the creation of a generalized set of automatic matching algorithms. Accordingly, the goal of this work is to present an evaluation framework for automatic matching algorithms as well as to model the uncertainty of such a process.

To reduce manual intervention, many suggestions have been made over the last two decades – both by scholars and by industry – to bring about a higher level of automation of the matching process among schemata and to reduce semantic mismatch problems. A useful classification of the various solutions can be found in [56]. Of the various dimensions presented there, we focus on those categories that highlight the algorithmic aspect of the problem. The proposed solutions can be grouped into four main approaches. The **first** approach recommends adoption of information retrieval techniques. Such techniques apply approximate, distance-based (e.g. edit distance [41] as proposed in [44]), matching techniques, thus overcoming the inadequacy of exact, "keyword-based" matching. This approach is based on the presumption that similar attribute names represent semantic similarity. Attribute names are rarely, however, given in an explicit form that yields good matchings. Furthermore, they need to be complemented by either a lengthier textual description or explicit thesaurus, which mandates greater human intervention in the process. Protégé utilizes this method in the PROMPT (formerly SMART) algorithm, a semiautomatic matching algorithm that guides experts through ontology matching and alignment [27,28].

A **second** approach to the matching of schemata involves the adoption of machine learning algorithms that create a mapping between two attributes based on the similarity among their associated values. Most existing approaches (e.g., GLUE [14] and Autoplex [6]) adopt some form of a Bayesian classifier [16,40]. Pursuant to this approach, mappings are based on classifications with the greatest posterior probability, given data samples. Another method that can be utilized for schema matching is that of grammatical inferences [32,51,57]. This method, utilized in the area of natural language processing, involves the inference of a grammar G, as a regular expression, from a set of examples of a language $L(G)$. Machine learning was recognized as an important aspect of reasoning about mappings in [42].

Third, several scholars have suggested the use of graph theory techniques to identify similarity among schemata, represented in the form of either a tree or a graph [9,52,60]. For example, the TreeMatch algorithm [43] utilizes XML DTD's tree structure in evaluating the similarity of leaf nodes by estimating the similarity of their ancestors. Also, the work of Valtchev and Euzenat [63] (aimed at automatic classification) applies a similarity measure in which the dissimilarity between objects is measured in terms of their distance from a common class in a given classification scheme [44].

A **fourth** approach involves a hybrid of matching techniques from the three approaches given above. Under this approach, a weighted sum of the output of algorithms in these three categories serves to specify the similarity of any two schema elements. Cupid [43] and OntoBuilder [49] are two models that support the hybrid approach. Also, the research into rank aggregation methods [18,21] can be applied in this context to combine the results of various matching algorithms.

A few other systems (MOMIS [5], DIKE [55], and Clio [48], to name a few) aim at resolving semantic heterogeneity in heterogeneous databases. However, these models assume manual intervention on a grand scale.

There is sparse academic literature on the appropriate evaluation tool for proposed algorithms and matching methods in this area (an initial effort is available in [42]; see below). The proposed framework identifies in a systematic manner the shortcomings of automatic schema matching. In particular, it models schema matching as a process with uncertain outcomes and identifies sufficient conditions for effective schema matching as a feedback for improving matching algorithms.

2.2 Ontology design

The second body of literature the study draws upon focuses on ontology design. Ontologies have been widely accepted as the model of choice for modeling heterogeneous data sources by various communities including databases [15,35,49] and knowledge representation [28], to name two.

The area of information science has an extensive body of literature and practice on ontology construction using tools such as thesauri and on terminology rationalization and matching of different ontologies [1,58,61,65]. Other works, such as the DOGMA project [35,62], provide an engineering approach to ontology management. Finally, scholars in the area of knowledge representation have studied ontology interoperability, resulting in systems such as Chimaera [45], Protégé [28] (together with Prompt [26], an interactive algorithm for ontology merging), and RDFT [53], a mapping metaontology that maps business constructs such as events, documents, and vocabularies using such standards as WSDL and PSL.

The body of research aiming at matching schemata by using ontologies has traditionally focused on interactive methods, requiring sometimes massive human intervention.However, the new vision of the Semantic Web necessitates the minimization of human intervention, replacing it with syntactic similarity measures to approximate semantic matching. Thus, recent works (e.g., [15,49]) have looked into automatic semantic reconciliation using ontologies. It had been observed before that automatic matching may carry with it a degree of uncertainty since "the syntactic representation of schemas and data do not completely convey the semantics of different databases" [47]. In this work, we analyze and model fully automated semantic reconciliation, allowing a certain level of uncertainty in the matching outcome.

Several ontological languages were proposed to support the Semantic Web, including RDF/S, DAML+OIL, and OWL, as well as proposals for embedded semantics (e.g., [64]). Such tools add semantics at a metadata level through the use of

constructs such as constraints. As such, these models are concerned less with adequate structures and more with giving, through relationships, appropriate interpretation to terms. The work we present is model independent (although our experiments were performed on HTML forms). Generally speaking, however, the proposed framework can maintain mapping similarities from any term pairwise matching, be it based on naming conventions, structural constraints, or semantic constraints. We refer the interested reader to [30] for techniques for identifying ontological constructs and utilizing them in the context of semantic reconciliation.

2.3 Modeling alternatives

A recent work on representing and reasoning about mappings between domain models was presented in [42]. This work provides a model representation and inference analysis. Managing uncertainty was recognized as the next step on the research agenda in this area and was left open for a future research. Our work fills this gap in providing a model that represents the uncertainty (as an imprecision measure) in the matching process outcome.

In [46], a model for estimating information loss in a matching process was introduced. The model computes precision and recall of substitutions of terms in a generalization-specialization hierarchy, using both intentional and extensional measures. These metrics (and their combination, as suggested in [46]) serve as alternatives to the μ-confidence proposed in this paper. However, no value-of-information analysis was reported. That is, no evaluation of the correspondence of these measures to the "goodness" of the mapping, as perceived by an expert, are available. Our work shows that μ-confidence can be correlated with mapping quality.

Our approach was inspired by the works of Fagin and Wimmers [22] and Fagin [20], who proposed a method of combining answers to queries over different data sources using simple fuzzy set theory concepts and a method for allowing users to set weights to different parts of their queries. This work extends imprecision to metadata (and thus makes it a viable resource for Semantic Web-related algorithms) as well and identifies a family of mappings for which imprecision calculations are meaningful.

An alternative to the fuzzy sets framework exists in the form of probabilistic methods (e.g., [19,39]). A probabilistic-based approach assumes that one has incomplete knowledge about the portion of the real world being modeled. However, this knowledge can be encoded as probabilities about events. The fuzzy approach, on the other hand, aims at modeling the intrinsic imprecision of features of the modeled reality. Therefore, the amount of knowledge at the user's disposal is of little concern. Our choice, in addition to philosophical reasoning, is also based on pragmatic reasoning. Probabilistic reasoning typically relies on event independence assumptions, making correlated events harder, if not impossible, to assess. Our approach is supported by the results presented in [17] in which a comparative study of the capabilities of probability and fuzzy methods is presented. This study shows that probabilistic analysis is intrinsically more expressive than fuzzy sets. However, fuzzy methods demonstrate higher computational efficiency.

3 Preliminaries

In this section we present two families of operators, namely, triangular norms (Sect. 3.1) and fuzzy aggregate operators (Sect. 3.2), and compare their properties. Operators from both families are typically used in fuzzy-based applications to combine various fuzzy membership degrees. Since the study of different ways of combining similarities is crucial to this work, we provide a brief introduction to their main properties.

3.1 Triangular norms

The $\min$ operator is the most well-known representative of a large family of operators called *triangular norms* (t-norms for short), routinely deployed as interpretations of fuzzy conjunctions (see, for example, the monographs [33,38]). In the following we define t-norms and discuss their relevant properties. We refer the interested reader to [37] for an exhaustive treatment of the subject.

A *triangular norm* $T : [0,1] \times [0,1] \rightarrow [0,1]$ is a binary operator on the unit interval satisfying the following axioms for all $x, y, z \in [0,1]$:

$T(x,1) = x$ (boundary condition),
$x \leq y$ implies $T(x,z) \leq T(y,z)$ (monotonicity),
$T(x,y) = T(y,x)$ (commutativity),
$T(x,T(y,z)) = T(T(x,y),z)$ (associativity).

The following t-norm examples are typically used as interpretations of fuzzy conjunctions:

$Tm(x,y) = \min(x,y)$ (minimum t-norm)
$Tp(x,y) = x \cdot y$ (product t-norm)
$Tl(x,y) = \max(x+y-1,0)$ (Lukasiewicz t-norm).

It is worth noting that Tm is the only idempotent t-norm. That is, $Tm(x,x) = x$.[3] This becomes handy when comparing t-norms with fuzzy aggregate operators (Sect. 3.2). It can be easily proven ([33]) that

$$Tl(x,y) \leq Tp(x,y) \leq Tm(x,y) \tag{1}$$

for all $x, y \in [0,1]$.

All t-norms over the unit interval can be represented as a combination of the triplet (Tm, Tp, Tl) (see [33] for a formal presentation of this statement). For example, the Dubois-Prade family of t-norms T^{dp}, also used often in fuzzy set theory and fuzzy logic, is defined using Tm, Tp, and Tl as:

$$T^{dp}(x,y) = \begin{cases} \lambda \cdot Tp(\frac{x}{\lambda}, \frac{y}{\lambda}) & (x,y) \in [0,\lambda]^2 \\ Tm(x,y) & \text{otherwise.} \end{cases}$$

3.2 Fuzzy aggregate operators

The $average$ operator belongs to another large family of operators termed *fuzzy aggregate operators* [38]. A fuzzy aggregate

[3] For a binary operator f, idempotency is defined to be $f(x,x) = x$ (similar to [38], p. 36).

operator $H : [0,1]^n \rightarrow [0,1]$ satisfies the following axioms for every $x_1, \ldots, x_n \in [0,1]$:

$$H(x_1, x_1, \ldots, x_1) = x_1 \text{ (idempotency)}, \quad (2)$$

for every $y_1, y_2, \ldots, y_n \in [0,1]$ such that $x_i \leq y_i$,

$$H(x_1, x_2, \ldots, x_n) \leq H(y_1, y_2, \ldots, y_n)$$

(increasing monotonicity), (3)

H is a continuous function. (4)

Let $\bar{x} = (x_1, \ldots, x_n)$ be a vector such that for all $1 \leq i \leq n, x_i \in [0,1]$ and let $\bar{\varpi} = (\varpi_1, ..., \varpi_n)$ be a weight vector that sums to unity. Examples of fuzzy aggregate operators include the $average$ operator $Ha(\bar{x}) = \frac{1}{n}\sum_1^n x_i$ and the $weighted\ average$ operator $Hwa(\bar{x}, \bar{\varpi}) = \bar{x} \cdot \bar{\varpi}$. Clearly, $average$ is a special case of the $weighted\ average$ operator, where $\varpi_1 = \cdots = \varpi_n = \frac{1}{n}$. It is worth noting that Tm (the $\min$ t-norm) is also a fuzzy aggregate operator due to its idempotency (its associative property provides a way of defining it over any number of arguments). However, Tp and Tl are not fuzzy aggregate operators.

T-norms and fuzzy aggregate operators are comparable using the following inequality:

$$\min(x_1, \ldots, x_n) \leq H(x_1, \ldots, x_n)$$

for all $x_1, \ldots, x_n \in [0,1]$ and function H satisfying Eqs. 2–4.

4 The framework

In this section we provide a formal framework for computing similarities among attribute (concept) sets based on fuzzy relations [38], as follows. A *fuzzy set* A over a domain $\mathcal{D}$ is a set characterized by a membership function $\delta_A : \mathcal{D} \rightarrow [0,1]$, where $\delta_A(a) = \mu$ is the fuzzy membership degree of the element a in A. In what follows we use μ^a to specify the elements of interest whenever it cannot be clearly identified from the context. Given domains $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n$ and their Cartesian product $\mathcal{D} = \mathcal{D}_1 \times \mathcal{D}_2 \times \cdots \times \mathcal{D}_n$, a *fuzzy relation* R over the domains $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_n$ is a fuzzy set of elements (tuples) of $\mathbf{D}$. $\mu^{d_1, d_2, \ldots d_n}$ represents the fuzzy membership degree of the tuple $(d_1, d_2, \ldots d_n)$ in R.

We next introduce confidence relations, which we use to compute similarity. Primitive confidence relations are introduced in Sect. 4.1, and Sect. 4.2 introduces compound confidence relations.

4.1 Primitive confidence relations

Given domains $\mathcal{D}$ and $\mathcal{D}'$, a *primitive confidence* relation is a fuzzy relation over $\mathcal{D} \times \mathcal{D}'$, denoted $\sim_\mu$, where μ (also annotated $\mu^{d,d'}$) is the membership degree of the pair $\langle d, d' \rangle$ in $\sim_\mu$ [denoted the *mapping confidence of* (d, d')]. A mapping confidence of a primitive confidence relation is computed using some distance metric among domain members. Some desirable properties of a primitive confidence relation are as follows.

Reflexivity: $\mu^{d,d} = 1$. Reflexivity ensures that the exact mapping receives the highest possible score (as in the case of two identical attributes, e.g., with the same name).

Symmetry: $\mu^{d,d'} = \mu^{d',d}$. Symmetry ensures that the order in which two schemata are compared has no impact on the final outcome.

Transitivity: $\mu^{d,d''} \geq \max_{d' \in \mathcal{D}'} \min \left[\mu^{d,d'}, \mu^{d',d''}\right]$. This type of transitivity is known as the *max-min transitivity* property (e.g., [38], p. 130). It provides a solid foundation for the generation of fuzzy equivalence relations. As an example, one may generate α-level equivalence, which contain all pairs whose confidence measure is greater than α. While being a desirable property, transitivity is hard to achieve, and therefore we shall concentrate on proximity relations (satisfying reflexivity and symmetry) instead. Such a relation may, at some α level, generate a partition of the domain, similarly to α-level equivalence.

Example 3 (Value mapping confidence). Consider two non-negative numeric domains $\mathcal{D} = \{0, 15, 30, 45\}$ and $\mathcal{D}' = \{0, 10, 20, 30, 40, 50\}$, both representing a fraction of an hour in which a car will be picked up. Assume that the mapping confidence of elements $d \in \mathcal{D}$ and $d' \in \mathcal{D}'$ is measured according to their Euclidean distance, normalized between 0 and 1:

$$\mu^{d,d'} = 1 - \frac{|d - d'|}{\max_{d_i \in \mathcal{D}, d'_j \in \mathcal{D}'}\{|d_i - d'_j|\}}. \quad (5)$$

Therefore, the mapping confidence of 15 (in $\mathcal{D}$) and 30 (in $\mathcal{D}'$) is 0.7.

This primitive confidence relation, with its associated $\mu^{d,d'}$ as defined in Eq. 5, is reflexive (since $d - d = 0$) and symmetric (since $|d - d'| = |d' - d|$) yet nontransitive, which makes it a proximity relation. As an example, consider a third domain $\mathcal{D}'' = \{0, 30\}$. For $d = 0$ and $d'' = 30$, $\mu^{d,d''} = 0.33$, yet $\max_{d' \in \mathcal{D}'} \min \left[\mu^{d,d'}, \mu^{d',d''}\right] = 1$ (e.g., for $d' = d = 0$). □

Example 4 (Attribute name mapping confidence). Let $\mathcal{A}$ and $\mathcal{A}'$ be two domains whose elements are attribute names. Let $\sim_{\mu_{attrname}}$ be a primitive confidence relation over $\mathcal{A} \times \mathcal{A}'$, where $\mu_{attrname}$ is termed the *attribute name mapping confidence measure*. The computation of attribute name mapping confidence is typically based on substring matching and is enhanced by the use of information retrieval techniques, such as dehyphenation [23] and stop term removal [25].

Example 2 suggests the use of the dehyphenation technique, combined with substring matching to compute $\mu_{attrname}$. The method proposed there can be described as follows:

$$\mu^{A,A'}_{attrname} = \frac{|A \cap A'|}{\max(|A|, |A'|)}, \quad (6)$$

where $|A \cap A'|$ stands for the length of the longest common substring (after preprocessing such as dehyphenation). It is worth noting that, as is often common in the database literature, we let A refer to both an attribute and its name. This primitive confidence relation, with its associated $\mu^{A,A'}_{attrname}$, as defined in Eq. 6 is reflexive, since for two identical attribute names (e.g., `PickUpMinutes` of the `AvisRental` and the `AlamoRental` relations) the size of the common substring is the whole attribute name, and therefore $\mu_{attrname} = 1$. Also, it is symmetric since

$|A \cap A'| = |A' \cap A|$ and $\max(|A|, |A'|) = \max(|A'|, |A|)$. However, it is nontransitive, which, again, makes it a proximity relation. As an example, consider three schemata with one attribute each, e.g., `FirstChoice`, `PrimaryChoice`, and `PrimarySelection`. While `FirstChoice` matches `PrimaryChoice` with $\mu_{attrname} = 0.46$ and `PrimaryChoice` matches `PrimarySelection` with $\mu_{attrname} = 0.44$, matching `FirstChoice` with `PrimarySelection` results in $\mu_{attrname} = 0$.

Another method of computing attribute name confidence divides the length of the longest common substring by the length of the first (or, alternatively, the second) attribute name, given by

$$\mu^{A,A'}_{attrname} = \frac{|A \cap A'|}{|A|}.$$

Clearly, such a measure is asymmetric. For example, `FirstChoice` matches `PrimaryChoice` with $\mu_{attrname} = 0.55$, yet `PrimaryChoice` matches `FirstChoice` with $\mu_{attrname} = 0.46$. □

By formalizing confidence measures, one can better analyze the properties of matching techniques. For example, consider the three attributes `FirstChoice`, `PrimaryChoice`, and `PrimarySelection`, discussed in Example 4. This example highlights the importance of transitivity. The three attributes seem to be semantically similar, referring to some top priority option, and therefore in the presence of three schemata one would be interested in placing the three together in a single equivalence class. However, nontransitivity prevents the substring matching technique from achieving such a mapping. Many of the confidence relations we have encountered are proximity relations, which increase the complexity of the matching process. In particular, with the introduction of a new schema it does not suffice to perform the matching process with a single representative schema (which can be efficiently performed using simple matrix multiplication techniques) from the set of known schemata. Rather, the matching process should be performed in a pairwise fashion with every schema in the schema set.

4.2 Compound confidence relations

Compound confidence relations are fuzzy relations as well. Yet they use confidence measures (either primitive or compound) to compute new confidence measures. In this section we introduce three examples of compound confidence relations and discuss their properties.

Example 5 (Domain confidence relation). Example 3 suggests a method for computing value confidence measure for nonnegative numeric domains. Next, we can compute the mapping confidence of two such domains based on the mapping confidence of their values. Let $\mathcal{D}$ and $\mathcal{D}'$ be two domains taken from a domain whose elements are themselves domains. Let μ_{dom} be a function termed the *domain mapping confidence measure*. Then $\sim_{\mu_{dom}}$ is a *domain mapping confidence* relation. μ_{dom} is a function of the mapping confidence of every pair of elements from $\mathcal{D}$ and $\mathcal{D}'$. For example, one may compute μ_{dom} as

$$\mu^{\mathcal{D},\mathcal{D}'}_{dom} = \min_{d \in \mathcal{D}, d' \in \mathcal{D}'} \left(\mu^{\mathcal{D},d'}, \mu^{\mathcal{D}',d} \right), \tag{7}$$

where for all $d' \in \mathcal{D}'$, $\mu^{\mathcal{D},d'} = \max_{d \in \mathcal{D}} \left(\mu^{d,d'} \right)$ and for all $d \in \mathcal{D}$, $\mu^{\mathcal{D}',d} = \max_{d' \in \mathcal{D}'} \left(\mu^{d,d'} \right)$. That is, each value in $\mathcal{D}$ is matched with the "best" value in $\mathcal{D}'$, and vice versa, and the strength of μ_{dom} is determined by the strength of the "weakest link". Our use of $\min$ and $\max$ is in line with fuzzy logic conventions, where $\max$ is interpreted as disjunction and $\min$ is interpreted as conjunction. We shall discuss alternative operators in Sect. 5.2, providing constraints on possible operator selections.

As a concrete example, consider $\mathcal{D}$ and $\mathcal{D}'$ of Example 3. Computing $\mu^{\mathcal{D},\mathcal{D}'}_{dom}$ according to Eq. 7 yields a matching of 0 with 0, 10 and 20 with 15, etc. $\mu^{\mathcal{D},\mathcal{D}''}_{dom} = 0.9$, since each element in $\mathcal{D}'$ has a corresponding element in $\mathcal{D}$, which is at most 5 min away (and $1 - \frac{5}{50} = 0.9$).

Proposition 1. *The domain mapping confidence relation is a proximity relation.*

Proof. We shall now show that Eq. 7 is reflexive and symmetric.

Reflexivity: From the fact that $\mathcal{D} = \mathcal{D}'$ one has that for all $d' \in \mathcal{D}$,

$$\begin{aligned} \mu^{\mathcal{D},d'} &= \max_{d \in \mathcal{D}} \left(\mu^{d,d'} \right) \\ &= \mu^{d,d} \\ &= 1. \end{aligned}$$

Therefore,

$$\mu^{\mathcal{D},\mathcal{D}}_{dom} = \min_{d \in \mathcal{D}, d' \in \mathcal{D}} \left(\mu^{\mathcal{D},d'}, \mu^{\mathcal{D},d} \right) = 1.$$

Symmetry: We show that $\mu^{\mathcal{D},\mathcal{D}'}_{dom} = \mu^{\mathcal{D}',\mathcal{D}}_{dom}$:

$$\begin{aligned} \mu^{\mathcal{D},\mathcal{D}'}_{dom} &= \min_{d \in \mathcal{D}, d' \in \mathcal{D}'} \left(\mu^{\mathcal{D},d'}, \mu^{\mathcal{D}',d} \right) \\ &= \min_{d' \in \mathcal{D}', d \in \mathcal{D}} \left(\mu^{\mathcal{D}',d}, \mu^{\mathcal{D},d'} \right) \\ &= \mu^{\mathcal{D}',\mathcal{D}}_{dom}. \end{aligned}$$

In general, the computation of μ_{dom} needs to consider all nonzero similarities between elements of $\mathcal{D}$ and $\mathcal{D}'$. Therefore, the computation complexity of μ_{dom} is of $O(|\mathcal{D}| \times |\mathcal{D}'|)$, where $|\mathcal{D}|$ and $|\mathcal{D}'|$ are the cardinalities of $\mathcal{D}$ and $\mathcal{D}'$, respectively.[4] Such complexity becomes tedious for big domains. For certain special cases, however, domain confidence can be computed at a much lower cost. For example, when computing Eq. 7 for sorted numeric domains using Euclidean distance as the distance metric, each element in one domain needs to be matched with at most two elements in the other domain (using a variation of the merge-sort algorithm), reducing the overall complexity of the process to $O(|\mathcal{D}| + |\mathcal{D}'|)$. Also, if one domain has even a single value that cannot be matched with any value in the other domain (e.g., by adding a text value "Choose from list" to one of two numeric domains), then, using Eq. 7, $\mu^{\mathcal{D},\mathcal{D}'}_{dom} = 0$.

[4] This analysis assumes domains with a finite number of elements.

Table 1. Computing attribute-set similarity measure

Attribute pair	$\mu_{attrname}$	μ_{dom}	μ_{att}
RentalNo,RentalNo	1	1	1
PickUpLocationCode,PickUpLocation	0.78	1	0.89
PickUpDate,PickUp-Date	1	1	1
PickUpHour,PickUpHour	1	1	1
PickUpMinutes,PickUpMinutes	1	0.9	0.95
ReturnDate,DropoffDate	0.36	1	0.68
ReturnHour,DropoffHour	0.36	1	0.68
ReturnMinutes,DropoffMinutes	0.5	0.9	0.7
Price,Price	1	1	1
			0.88

Other methods for computing domain confidence measure have been proposed in the literature. For example, in [63], a method for computing domain confidence based on optimal weighted bipartite graph was proposed. Such a method minimizes the dissimilarity measure, at the expense of partial mapping, where there exist nonmapped values in case of different domain cardinalities. □

Example 6 (Attribute mapping confidence relation). In [49], attribute mapping confidence is determined as a combination of attribute name mapping confidence ($\mu_{attrname}$) and the mapping confidence between the corresponding attribute domains, as presented in Example 5 (μ_{dom}). Therefore, given two attributes A and A', with domains $\mathcal{D}$ and $\mathcal{D}'$, respectively, the *attribute confidence measure* of A and A', denoted μ_{att}, is a function $\mu_{att}^{A,A'} = h_1(\mu_{attrname}^{A,A'}, \mu_{dom}^{\mathcal{D},\mathcal{D}'})$.

Consider the attributes `PickUpMinutes` of the `AvisRental` and the `AlamoRental` relations. Since both relations use the same attribute name, $\mu_{attrname} = 1$ using substring matching. Their corresponding domains are $\{0, 15, 30, 45\}$ and $\{0, 10, 20, 30, 40, 50\}$, and, as shown in Example 5, $\mu_{dom} = 0.9$. Assuming $\mu_{att} = average(\mu_{attrname}, \mu_{dom})$, one has that $\mu_{att} = 0.95$. Comparing `PickUpMinutes` of the `AvisRental` relation and `DropoffMinutes` of the `AlamoRental` relation yields $\mu_{attrname} = 0.5$ and $\mu_{dom} = 0.9$. Therefore, $\mu_{att} = 0.7$. □

Example 7 (Schema mapping confidence). Given two attribute sets, $\mathcal{A}$ and $\mathcal{A}'$, a *schema mapping* F from $\mathcal{A}$ to $\mathcal{A}'$ is a set of $|\mathcal{A}|$ pairs (A, A') such that $A \in \mathcal{A}$, $A' \in \mathcal{A}' \cup \{null\}$, and $A' = F(A)$. A mapping to a null value represents no mapping. $\mathcal{A} \sim_\mu \mathcal{A}'$ denotes the *schema mapping confidence* of F. The *schema mapping confidence measure* μ^F is a function $\mu^F = h_2(\mu_{att}^{A,A'} | (A, A') \in F)$.

In Example 2 we have provided a possible set of schema containment rules. Using this set, we have selected a mapping F, given in Table 1. It is worth noting that this mapping is only one among many ($n!$ for $1:1$ matching). The table provides the computation of $\mu_{attrname}$ using dehyphenation and substring matching (see Example 2) and the computation of μ_{dom} using the $\min$ function over the pairwise element confidence (see Example 5). μ_{att} is computed using the $average$ function as the computation operator. Computing μ^F by averaging over $\mu_{att}^{A,A'}$ of all the pairs (A, A') in F yields $\mu^F = 0.88$.

Generally speaking, a mapping can be $1:1$ (in which case the mapping becomes a $1:1$ and onto function), $1:n$ (in which an attribute from the scope can be mapped into multiple attributes in the domain, either as is or by splitting an attribute value from the scope and assigning different attributes in the domain with subvalues), or $n:1$ (see [7] for more details). Typically, one would be interested in a *best mapping*, i.e., a mapping with the highest score of all possible mappings. Methods for computing the best mapping depend on the type of mapping. For a $1:1$ matching, algorithms for identifying the best mapping typically rely on weighted bipartite graph matching [31]. In Sect. 5.2 we formally justify the use of such algorithms. □

4.3 Discussion

The examples in this section define value, attribute name, domain, attribute, and schema mapping confidence measures. Extension to this basic model can also be attained. For example, advanced works such as [48] generate mappings in which attributes are mapped through complex structures, including $n:1$ mappings. In [48], a scenario is introduced in which attribute `sal` from relation `professor` is combined with attributes `hrrate` and `workson` from relation `payrate`, to compute the attribute `sal` in `personnel` relation from the target schema. The assignment of confidence to such mapping can be defined in a variety of methods. Once such a measure is presented, it can be used in computing mapping confidence using the method in Example 7. It is worth noting that, in the absence of any restrictions on the mapping cardinality, computing $n:1$ mappings may require computing 2^n pairwise confidence measures, which is obviously intractable. As another example, extending the matching process to include graph theory methods, as suggested in [30,43], involves extending $\mu_{att}^{A,A'}$ by adding a third parameter that indicates the confidence as derived from the graph algorithm.

In a heterogeneous databases environment, it has been recognized that mapping of a single relation in a global schema to a single relation in a source requires a high level of uniformity in how data are represented in the sources and how they are viewed in the global schema. To overcome such structural heterogeneity, structure-oriented algorithms (e.g., the TreeMatch algorithm of Cupid and the precedence algorithm in OntoBuilder) were proposed. In Sect. 6 we experiment with a representative algorithm that utilizes structural information.

5 Monotonic mappings: measuring matching quality

In this section we aim at modeling the relationship between a choice of a schema mapping, based on similarity of attributes, and a choice of a schema mapping, as performed by a human expert. As we empirically show in Sect. 6, the more correlated these mappings are, the more effective would be an automatic mapping process. Therefore, monotonicity is aimed at ensuring that the exact mapping can be found by iterating over a small set of mappings (a single mapping, the best mapping, in the case of strict monotonicity). Section 5.1 provides the basic definitions of the monotonicity notion. A discussion of monotonicity properties is given in Sect. 5.2. Finally, we provide

two weaker notions of monotonicity that are explored further in our empirical analysis (Sect. 5.3).

5.1 Monotonicity

To compare the effectiveness of various choices of mappings and operators, we introduce the notion of mapping *imprecision*, which follows common IR practice for retrieval effectiveness (e.g., [24]). First, we define mapping difference as follows.

Definition 1 (Mapping difference). *Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\mathcal{A}' = \{A'_1, \ldots, A'_n\}$ be attribute sets of cardinality n. Also, let F and G be two schema mappings over $\mathcal{A}$ and $\mathcal{A}'$ and let $A_i \in \mathcal{A}$ be an attribute. F and G* differ *on A_i if $F(A_i) \neq G(A_i)$. $\mathcal{D}^{F,G}$ denotes the set of attributes of $\mathcal{A}$ on which F and G differ.*

Imprecision is defined next simply by counting how many arguments of two schemata F and G do not coincide.

Definition 2 (Imprecision). *Let F and G be two schema mappings over two attribute sets of cardinality n, $\mathcal{A}$ and $\mathcal{A}'$. Assume that there are $m \leq n$ attributes in $\mathcal{A}$ on which F and G differ. Then G is $\frac{m}{n}$-*imprecise *with respect to F and F is $\frac{m}{n}$-*imprecise *with respect to G. We denote by $i_{F,G}$ the imprecision level.*

Example 8 (Imprecision). A mapping between `AvisRental` and `AlamoRental` is given by the containment rules of Example 1 and Table 1. Consider a mapping that varies from the one presented in Table 1 by associating `PickUpDate` with `DropoffDate` and `ReturnDate` with `Pickup-Date`. Their attribute confidence scores, μ_{att}, are 0.68 and 0.7, respectively. Such mapping attains a lower mapping confidence degree than the mapping presented in Table 1, where `PickUpDate` is matched with `Pickup-Date` (confidence of 1) and `DropoffDate` is matched with `ReturnDate` (confidence of 0.68). The two mappings are $\frac{2}{9}$-imprecise with respect to one another, according to Definition 2. □

It is worth noting that imprecision, while normalized to be in $[0, 1]$, cannot accept all possible values in this range. Therefore, for an attribute set of n attributes, one can have exactly n imprecision categories.

Definition 3 (Confidence fortification). *Let F, G, and H be mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$. G and H are* confidence fortification *on an attribute $A \in \mathcal{A}$ with respect to F if $i_{F,G} < i_{F,H}$ implies $\mu^{A,G(A)} > \mu^{A,H(A)}$. $\mathcal{M}^{G,H}$ denotes the set of attributes of $\mathcal{A}$ on which G and H are confidence fortifying with respect to F.*

Example 9 (Confidence fortification). Example 8 has introduced a $\frac{2}{9}$-imprecise mapping by associating `PickUpDate` with `DropoffDate` and `ReturnDate` with `Pickup-Date`. Referring to this mapping as H and to the mapping of Example 1 as both F and G, G and H are confidence fortifying (with respect to F) on attribute `Pickup-Date`, since $0 = i_{F,G} < i_{F,H} = \frac{2}{9}$ and $1 = \mu_{att}^{A,G(A)} > \mu_{att}^{A,H(A)} = 0.68$. However, G and H are not confidence fortifying on attribute `ReturnDate` since $0.68 = \mu_{att}^{A,G(A)} \not> \mu_{att}^{A,H(A)} = 0.7$. □

Definition 4 (Benefit and cost). *Let G and H be schema mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$ such that $i_{F,G} < i_{F,H}$ with respect to some mapping F. Given a function h, the* benefit *of switching from H to G is defined as*

$$Benefit(G, H) = h_{A_k \in \mathcal{D}^{G,H} \cap \mathcal{M}^{G,H}} \left(\mu^{A_k, G(A_k)} - \mu^{A_k, H(A_k)} \right).$$

The cost *of switching from H to G is defined as*

$$Cost(G, H) = h_{A_k \in \mathcal{D}^{G,H} \setminus \mathcal{M}^{G,H}} \left(\mu^{A_k, H(A_k)} - \mu^{A_k, G(A_k)} \right).$$

$Benefit(G, H)$ represents the benefit of switching from H to G. $\mathcal{D}^{G,H} \cap \mathcal{M}^{G,H}$ represents those attributes over which G and H differ yet are confidence fortifying with respect to F. $Cost(G, H)$ represents the loss involved in switching from H to G. $\mathcal{D}^{G,H} \setminus \mathcal{M}^{G,H}$ represents those attributes over which G and H differ and that are not confidence fortifying with respect to F.

We shall next identify a family of "well-behaved" mappings as a quality measure for comparing various algorithms for schema matching. Assume that among all possible mappings between two attribute sets of cardinality n ($n!$ such mappings for $1:1$ matching), we choose one and term it the *exact mapping* (denoted by $\bar{F}$). The exact mapping corresponds to the best possible mapping, as conceived by a human expert.

Definition 5 (Monotonicity). *Let $\mathcal{F} = \{F_1, F_2, ..., F_m\}$ be a set of mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$. $\mathcal{F}$ is* monotonic with respect to $\bar{F}$ *if the following inequality holds for any pair $\{F_i, F_j\} \subseteq \mathcal{F}$ such that $i_{F_i} < i_{F_j}$:*

$$Benefit(F_i, F_j) > Cost(F_i, F_j). \quad (8)$$

i_{F_i} is a concise representation of $i_{\bar{F},F_i}$ and is used whenever imprecision is computed with respect to the exact mapping. Intuitively, the more imprecise a matching is with respect to a given exact mapping $\bar{F}$, the lower its corresponding confidence measure would be. Each term in $Benefit(F_i, F_j)$ adds to the overall confidence, yet the attributes that participate in computing $Cost(F_i, F_j)$ reduce the overall confidence by switching from F_j to F_i. If the benefit of switching from F_j to F_i surpasses the cost for all pairs $\{F_i, F_j\} \subseteq \mathcal{F}$ such that $i_{F_i} < i_{F_j}$, we consider the set to be monotonic. If the exact mapping is chosen from among monotonic mappings, then the following holds: if $\bar{F} \in \mathcal{F}$ and $\mathcal{F}$ is monotonic, then $\bar{F}$'s overall confidence measure is greater than the overall confidence measures of $\frac{i}{n}$-imprecise mappings in $\mathcal{F}$ ($i > 0$), even if such mappings yield better confidence measures on some attribute pairs.

Example 10 (Monotonic mappings). Consider the case study, as presented in Example 1, and the exact mapping between `AvisRental` and `AlamoRental` as given in Table 1 (defining h as the $average$ function). Using mapping confidence based on domain and attribute confidence measures, we have grouped the possible mappings between `AvisRental` and `AlamoRental` according to their level of imprecision. Figure 1 provides the highest, average, and

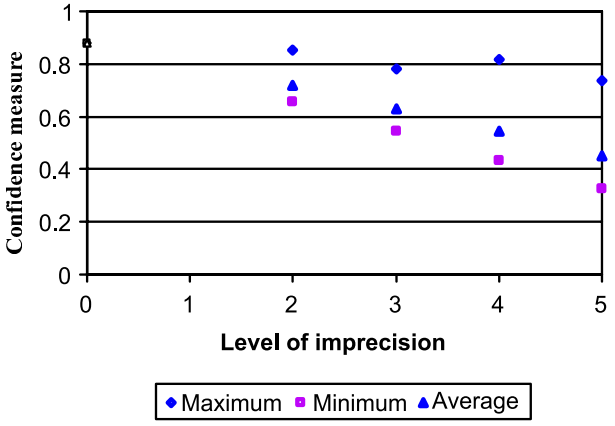

Fig. 1. Confidence vs. precision in the case study

lowest confidence measure of mappings for each level of imprecision. The figure demonstrates declining confidence measure (on average) as the imprecision increases. Nevertheless, this set of possible mappings is not monotonic. For example, consider the $\frac{3}{9}$-imprecise mapping, in which `RentalNo` is mapped into `PickUpHour`, `PickUpHour` is mapped into `Price`, and `Price` is mapped into `RentalNo`. The confidence measure of this mapping is 0.54. Consider now a $\frac{4}{9}$-imprecise mapping in which `PickUpLocationCode` is mapped into `PickUp-Date`, `PickUpDate` is mapped into `PickUpMinutes`, `PickUpMinutes` is mapped into `PickUpHour`, and `PickUpHour` is mapped into `PickUpLocation`. The confidence measure of this mapping is 0.55, slightly higher than a $\frac{3}{9}$-imprecise mapping. □

5.2 Monotonicity properties

If h is defined as a weighted average and the schema mapping confidence is computed using a weighted average as well, monotonicity can be specified in simpler terms, as the theorem below shows.

Theorem 1. *Let $\mathcal{F}$ be a monotonic set of mappings with respect to $\bar{F}$, using $h = Hwa$ (weighted average), and let $\{F_i, F_j\} \in \mathcal{F}$ be mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$ with imprecision i_{F_i} and i_{F_j}, respectively, such that $i_{F_i} < i_{F_j}$. If the schema mapping confidence measure is computed using the Hwa operator yielding, respectively, μ^{F_i} and μ^{F_j}, then $\mu^{F_i} > \mu^{F_j}$.*

Proof. The mappings F_i and F_j are part of a monotonic set $\mathcal{F}$, then by Definition 3 (using $h = Hwa$), and since $i_{F_i} < i_{F_j}$, the following inequality holds:

$$\sum_{A_k \in \mathcal{D}^{F_i,F_j} \cap \mathcal{M}^{F_i,F_j}} \left(\varpi_k \left(\mu_{att}^{A_k,F_i(A_k)} - \mu_{att}^{A_k,F_j(A_k)} \right) \right) > \sum_{A_k \in \mathcal{D}^{F_i,F_j} \setminus \mathcal{M}^{F_i,F_j}} \left(\varpi_k \left(\mu_{att}^{A_k,F\,j(A_k)} - \mu_{att}^{A_k,F_i(A_k)} \right) \right). \quad (9)$$

Since $0 \leq \mu_{att} \leq 1$, this implies (after the $\mu_{att}^{A_k,F_i(A_k)}$ terms on the right side are swapped with the $\mu_{att}^{A_k,F_j(A_k)}$ terms on the left side of Inequality 9) that

$$\sum_{A_k \in \mathcal{D}^{F_i,F_j}} \varpi_k \mu_{att}^{A_k,F_i(A_k)} > \sum_{A_k \in \mathcal{D}^{F_i,F_j}} \varpi_k \mu_{att}^{A_k,F_j(A_k)}. \quad (10)$$

Since the confidence measures of the attributes over which mapping F_i and F_j do not differ are equal, we adjoin them to both sides of Inequality 10 and obtain

$$\sum_{A_k \in \mathcal{A}} \varpi_k \mu_{att}^{A_k,F_i(A_k)} > \sum_{A_k \in \mathcal{A}} \varpi_k \mu_{att}^{A_k,F_j(A_k)}. \quad (11)$$

Since we use Hwa for combining attribute confidence measures, then

$$\mu^{F_i} > \mu^{F_j}. \quad (12)$$

Theorem 1 requires that confidence measures are combined using the Hwa (weighted average) operator. Also, the theorem requires that the same operator be utilized for computing the benefit and the cost. It is interesting to note that this property does not hold for all operators. Consider, for example, the $\min$ operator. Consider further two attribute sets, $\{a_1, a_2\}$ and $\{a'_1, a'_2\}$, with the following attribute mapping confidence:

	a'_1	a'_2
a_1	0.5	0.8
a_2	0.4	0.5

Let the exact mapping be a mapping such that a_1 is mapped with a'_1 and a_2 is mapped with a'_2. Using either Hwa or Tm, one has that the benefit of switching from the 1-imprecise ($\frac{2}{2}$-imprecise) mapping $\{\langle a_1, a'_2\rangle, \langle a_2, a'_1\rangle\}$ to the exact mapping is 0.1 and the cost is 0.3, and therefore the set of possible mappings is nonmonotonic by definition. Now, when schema mapping confidence is computed using Hwa, one has that the schema mapping confidence of the exact mapping (0.5) is lower than that of the 1-imprecise mapping (0.6), as expected. However, by using Tm, the schema mapping confidence of the exact mapping (0.5) is higher than that of the 1-imprecise mapping (0.4), which serves to show that Theorem 1 does not apply to the $\min$ operator.

Monotonicity is defined is such a way as to be strict in this paper. Relaxing it to nonstrict monotonicity (by requiring $Benefit(F_i, F_j) \geq Cost(F_i, F_j)$ when $i_{F_i} \leq i_{F_j}$) would have no practical benefit. Consider a situation in which all possible permutations of attribute mappings among two schemata yield the same confidence measure. Therefore, switching among the mappings yields 0 cost and 0 benefit. This means that the set of all schema mapping permutations is weakly monotonic, which provides little help in identifying the exact mapping. Strict monotonicity, however, ensures that the exact mapping is the one mapping to which the benefit of switching is (strictly) higher than the cost.

We now show that the use of weighted average is preferred over any t-norm operator to compute mapping confidence. For simplicity's sake we restrict our discussion to confidence measure, as defined using value confidence measure and attribute name confidence measure. The following result can be easily generalized to any confidence measure method.

We denote by $X_1X_2X_3$ a particular selection of operators for computing domain confidence measure (X_1), attribute confidence measure (X_2), and mapping confidence measure (X_3).

For example, $TmHaHa$ represents the particular operator selection, as suggested throughout the examples in Sect. 4. We next show that, in most cases, a selection of type X_1X_2Ha is superior to any selection of type $X_1X_2T_3$, where T_3 stands for any t-norm operator.

Definition 6 (Closely related attribute sets). *Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\mathcal{A}' = \{A'_1, \ldots, A'_n\}$ be attribute sets of cardinality n. $\mathcal{A}$ and $\mathcal{A}'$ are* closely related *if, for any mapping F over $\mathcal{A}$ and $\mathcal{A}'$, if $(A, A') \in F$, then $\mu_{att}^{A,A'} > 0$.*

Closely related attribute sets consist of attributes that may map well in various combinations. Considering the case study presented in this paper, the attribute sets of `Avis` and `Alamo` are not closely related. For example, a mapping of `Price` in `Avis` to any attribute but `Price` in `Alamo` yields 0 confidence measure. We next present a proposition arguing that t-norms are not suitable for modeling attribute sets that are not closely related.

Proposition 2. *Let $\mathcal{A} = \{A_1, \ldots, A_n\}$ and $\mathcal{A}' = \{A'_1, \ldots, A'_n\}$ be attribute sets of cardinality n. If $\mathcal{A}$ and $\mathcal{A}'$ are **not** closely related, any selection of operators of type $X_1X_2T_3$ yields a nonmonotonic mapping set.*

Proof. $\mathcal{A}$ and $\mathcal{A}'$ are not closely related. Therefore, there exists an attribute pair (A, A') such that $A \in \mathcal{A}$ and $A' \in \mathcal{A}'$ and $\mu_{att}^{A,A'} = 0$. Let $\mathcal{F}$ be the set of all mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$ and let $\bar{F} \in \mathcal{F}$ be the exact mapping. Assume that $\mathcal{F}$ is monotonic.

1. $(A, A') \in \bar{F}$. Assume that $\mu^{\bar{F}}$ is computed using an operator selection of the type $X_1X_2T_3$. For $T_3 = \min$, $\mu^{\bar{F}} \leq 0$, since it cannot be higher than $\mu_{att}^{A,A'}$. Since $\mu^{\bar{F}} \geq 0$ by definition, one has that $\mu^{\bar{F}} = 0$. Using Eq. 1 and the property that any t-norm can be represented as a combination of Tl, Tp, and Tm, we can generalize that $\mu^{\bar{F}} = 0$ for any operator selection of type $X_1X_2T_3$. Consider now a $\frac{2}{n}$-imprecise mapping F. $\mu^F \geq 0 = \mu^{\bar{F}}$, which contradicts the monotonicity assumption.
2. $(A, A') \notin \bar{F}$. Therefore, there exist attribute pairs $\{(A, A''), (A^*, A')\} \in \bar{F}$. Let F be a mapping that differs from $\bar{F}$ by replacing $\{(A, A''), (A^*, A')\}$ with $\{(A, A'), (A^*, A'')\}$. Since there are exactly two attributes on which $\bar{F}$ and F differ, F is $\frac{2}{n}$-imprecise. Also, since $(A, A') \in F$, $\mu^F = 0$ (see part 1 above). Now, let G be some $\frac{3}{n}$-imprecise mapping. $\mu^G \geq 0 = \mu^F$, which contradicts the monotonicity assumption.

An immediate corollary to Proposition 2 relates to mappings using weighted bipartite graph matching. Given two attribute sets, $\mathcal{A}$ and $\mathcal{A}'$, one may construct a weighted bipartite graph $G = (V, E)$ such that $V = \mathcal{A} \cup \mathcal{A}'$ and $(v_i, v_j) \in E$ if $v_i \in \mathcal{A}$, $v_j \in \mathcal{A}'$. The weight function $\varpi : \mathcal{A} \times \mathcal{A}' \rightarrow [0, 1]$ is defined as $\varpi(v_i, v_j) = \mu_{att}^{v_i,v_j}$. The weighted bipartite graph matching algorithm yields a $1:1$ mapping F with maximum weight $\Omega^F = \sum_{(v_i,v_j)\in F} \varpi(v_i, v_j)$. Given that $\mathcal{A}$ and $\mathcal{A}'$ are attribute sets of cardinality n that are not closely related, and assuming a selection of operators of type X_1X_2Ha, such mapping yields $\mu^F = \frac{1}{n}\Omega^F$. Therefore, the use of weighted bipartite graph matching is equivalent to a selection of operators of type X_1X_2Ha, which yields results as good as any selection of operators of type $X_1X_2T_3$, and possibly better.

5.3 Other forms of monotonicity

If the exact mapping is chosen among monotonic mappings, then the following holds: if $\bar{F} \in \mathcal{F}$ and $\mathcal{F}$ are monotonic, then $\bar{F}$'s overall confidence measure is greater than the overall confidence measure of $\frac{i}{n}$-imprecise mappings in $\mathcal{F}$ $(i > 0)$, even if such mappings yield better confidence measure on some attribute pairs. If all one wishes to obtain is the ability to identify the exact mapping through the use of confidence, one needs a weaker notion of monotonicity, as defined next.

Definition 7 (Pairwise monotonicity). *Let $\mathcal{F} = \{F_1, F_2, ..., F_m\}$ be the set of all possible mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$. $\mathcal{F}$ is* pairwise monotonic with respect to $\bar{F}$ *if the following inequality holds for any $F_i \in \mathcal{F}$:*

$$Benefit(\bar{F}, F_i) > Cost(\bar{F}, F_i). \tag{13}$$

The set of all possible mappings of the case study (see Example 10) is monotonic with respect to the exact mapping. Finally, the following definition captures the intuition accompanying Fig. 1. While one cannot argue that F is monotonic, the figure clearly identifies a monotonic trend. The next definition formalizes this intuition using statistical terms.

Definition 8 (Statistical monotonicity). *Let $\mathcal{F} = \{F_1, F_2, ..., F_m\}$ be a set of mappings over attribute sets $\mathcal{A}$ and $\mathcal{A}'$ of cardinality n, and let $\mathcal{F}_1, \mathcal{F}_2, ..., \mathcal{F}_n$ be subsets of $\mathcal{F}$ such that for all $1 \leq i \leq n$, $F \in \mathcal{F}_i$ iff F is $\frac{i}{n}$-imprecise. We define M_i to be a random variable, representing the confidence measure of a randomly chosen $\frac{i}{n}$-imprecise mapping. $\mathcal{F}$ is* statistically monotonic with respect to $\bar{F}$ *if the following inequality holds for any $1 \leq i \leq j \leq n$:*

$$E(M_i) > E(M_j), \tag{14}$$

where $E(M)$ stands for the expected value of M.

In Sect. 6.2 we shall explore this property further by experimenting with various mappings and using statistical hypothesis validation.

6 Empirical analysis

This section presents empirical results, testing two different algorithms using the proposed framework. The first (dubbed *term algorithm*) involves word similarity and string matching and is similar to algorithms in other tools, such as Cupid and Protégé. The other algorithm (dubbed *combined algorithm*) combines string matching with value matching and two structural algorithms, involving composition and precedence. Both algorithms compare two schemata (in the relational model sense), yet the combined algorithm is also provided with semantically rich information hidden in the forms, such as the structure of the data and the time constraints as provided by business rules, to improve the matching process. Such information can be encoded in ontological tools such as RDF/S,

DAML+OIL, and OWL. Full discussion of these algorithms is given in [30].

The analysis we propose is aimed at verifying empirically the correlation between a confidence measure (generated by a given algorithm) on the one hand and monotonicity on the other hand, using imprecision level as the experimentation tool. The purpose of this analysis is not to determine the "best" algorithm for schema matching, nor is it aimed at generating cost-effectiveness measure in choosing one algorithm or the other. Nevertheless, it is worthwhile showing the benefits of the combined algorithm over the term algorithm, using an example in [30], and given here for completeness sake.

Example 11 (Term and combined algorithms). The example is concerned with automatic form filling by rewriting a query given in a given ontology to a newly introduced ontology. Consider the Delta Airlines reservation system (Fig. 2). The form contains two time fields, one for departures and the other for return. Due to bad design (or designer error), the departure time entry is named `dept_time_1` while return time is named `dept_time_2`. Both terms carry an identical label, `Time`, since the context can be easily determined (by a human observer, of course) from the positioning of the time entry with respect to the date entry. For the American Airlines reservation system (Fig. 2, right), the two time fields of the latter were not labeled at all (counting on the proximity matching capabilities of an intelligent human observer) and therefore were assigned, using composition by association, the label `Departure Date` and `Return Date`. The fields were assigned the names `departureTime` and `returnTime`. Term matching would prefer matching both `Time(dept_time_1)` and `Time(dept_time_2)` of Delta with `Return Date(returnTime)` of American (note that "dept" and "departure" do not match, either as words or as substrings). However, using the combined algorithm, and precedence matching in particular, the two time entries were correctly mapped.

All datasets were collected from real-world Web forms (see below). We describe the experiments set up in Sect. 6.1. Statistical monotonicity and pairwise monotonicity are discussed in Sects. 6.2 and 6.3, respectively. In Sect. 6.4 we present the relationships between the two monotonicity types.

6.1 Experiment setup

All experiments were conducted using an in-house tool named *OntoBuilder*,[5] which runs under the Java 2 JDK version 1.4 or greater. OntoBuilder supports an array of matching and filtering algorithms. Algorithm parameters (such as weights) are specified using an XML configuration file that can be edited using a user-friendly interface. OntoBuilder also provides an applet version with the same features as the standalone version and the added functionality that allows users to access and use it within a Web client.

We have analyzed 42 Web forms, from eight different domains, namely, flight reservation, hotel reservation, dating and matchmaking, newspaper search engines, resume forms, e-mail address registration, book search engines, and advanced forms of general-purpose search engines. For each Web form, we have automatically extracted a schema.[6] Web forms were combined into pairs from the same domain, and for each pair (21 all-in-all) we have applied both algorithms.

For each Web form pair, we have computed all attribute pairwise mappings $\mu_{att}^{A,A'}$, using each of the two algorithms separately. For each pair we have determined the exact mapping $\bar{F}$ and partitioned all possible permutations into imprecision levels with respect to $\bar{F}$. Given two schemata $\mathcal{S}$ and $\mathcal{S}'$, with n and m attributes, respectively, and assuming that n' attributes of $\mathcal{S}$ can be mapped correctly to n' attributes of $\mathcal{S}'$ (which necessitates that $n' \leq m$ since we assume a $1:1$ mapping), the number of possible mappings of attributes in $\mathcal{S}'$ into $\mathcal{S}$ is

$$ {}_mC_{n'} \cdot {}_nP_{n'} = \frac{m!}{(m-n')!n'!}\frac{n!}{(n-n')!}. $$

${}_mC_{n'}$ represents the number of combinations of choosing a sample of size n' attributes (without regard to order) from a set of m attributes. ${}_nP_{n'}$ represents the number of variations of choosing n' attributes from a set of n attributes. For the simplified case in which $m = n = n'$, the number of mappings is equivalent to the number of permutations of one of the attribute sets, that is, $n!$. Due to the enormous number of possible permutations, we have limited our experiments to subschemata with nine attributes each. The attributes were selected randomly from the original set of attributes, with the only restriction being that an exact mapping can be determined for all attributes in the subschemata. For generating the 9! permutations and classifying them into imprecision levels, we have utilized a Visual Basic macro, executing within a MS Excel XP worksheet. A matrix of 9×9 pairwise confidence measures (μ_{att}) served as input to the macro. The output included all possible mapping variations; for each we have computed $\mu^F = h_2(\mu_{att}^{A,A'} | (A, A') \in F)$, where h_2 is taken to be the $average$ function, following the discussion in Sect. 5.2, and i_F, the imprecision level as defined in Sect. 5.1.

6.2 Statistical monotonicity

In Sect. 5 we introduced three different notions of monotonicity. The strictest one requires (according to Theorem 1) that, given a set of all possible mapping permutations between two schemata, a sorted list of mappings (according to confidence measure) will satisfy a partial ordering according to imprecision level. That is, for monotonicity to hold, a mapping of an imprecision level $\frac{i}{n}$ will be associated with a confidence measure μ, which is higher than any confidence measure of a mapping from a higher imprecision level. Example 5 demonstrates how difficult it is to achieve monotonicity even in a toy example. The inherent uncertainty of the matching process generates variability that may cause imprecision sets to overlap in their confidence measures. Indeed, in all our experiments we have never come across such a set of monotonic mappings.

We shall defer a discussion on pairwise monotonicity to Sect. 6.3. In this section we focus on statistical monotonicity.

[5] http://www.cs.msstate.edu/~gmodica/Education/OntoBuilder/

[6] It is worth noting that the extraction process may involve multiple forms for a given Web site.

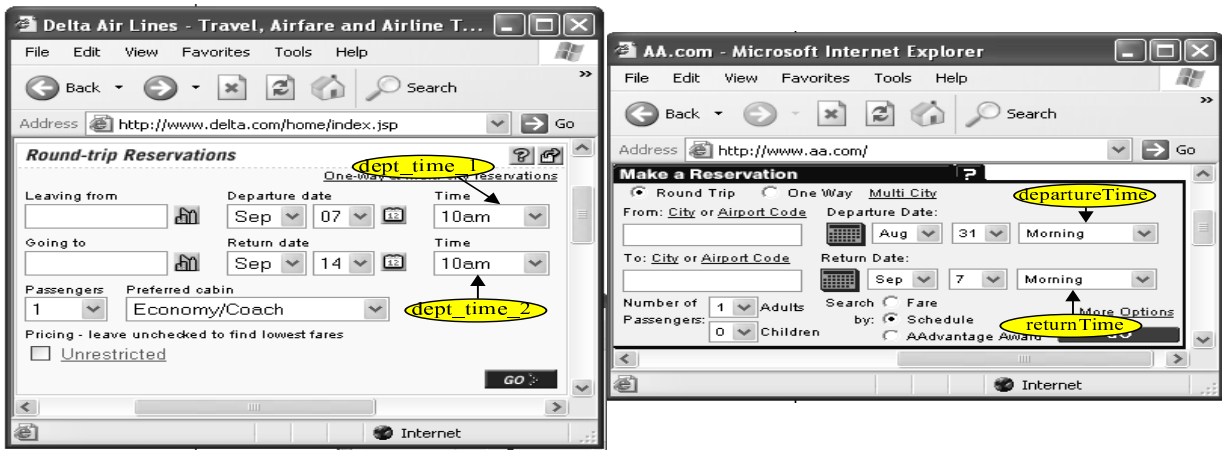

Fig. 2. AA vs. Delta

For statistical monotonicity to hold, we hypothesize that the confidence measure of a mapping is sensitive to the number of attributes on which the two schemata differ. That is, the confidence measure of a mapping is sensitive to the number of arguments that do not coincide. To evaluate this hypothesis, we examine how a confidence measure varies with imprecision level. To do so, we have performed linear regression analysis, focusing on the variability of the residual values around the regression line. We took special interest in the R^2 and X variable coefficient (the regression line gradient) statistics. The R^2 measure indicates the fraction of the total variability explained by the imprecision level. Plainly put, a high R^2 measure means that by separating the set of confidence measures into groups of imprecision levels, different groups have distinguished confidence measures.[7] A positive X variable coefficient is an indication of a positive correlation between imprecision level and confidence measure, while a negative X variable coefficient indicates negative correlation. Combined together, a negative X variable coefficient and a high R^2 measure indicate that imprecision is a major factor in determining the level of μ and that there is an inverse relation between the two. Such an indication is sufficient for ensuring statistical monotonicity.

The regression analysis was conducted using R GUI (a GNU project, based on the S language from Bell Labs), version 1.5.0. R is a programming environment for data analysis and graphics. To perform the regression analysis, we have collected a random representative sample of 500 instances from each imprecision level that has high number of permutations associated with it. By doing so, we ensure meaningful analysis, otherwise distorted by the sheer size of the analyzed set. Choosing a representative sample of the set of mappings allows an efficient execution of the analysis without adversely affecting the significance of the results.

Figure 3a illustrates a linear regression analysis of mapping "Absolute Agency" and "Adult Singles", from the dating and matchmaking domain, using the combined algorithm. For each mapping, the horizontal axis shows the imprecision level of a mapping, while the vertical axis provides the mapping confidence measure. The figure shows strong negative correlation between imprecision level and confidence measure. This conclusion is supported by the R^2 and X variable coefficient of the regression analysis. For this pair of sites, $R^2 = 0.97$, i.e., imprecision level explains, in this case, 97% of the original variability. X variable coefficient is -0.56. Therefore, we have sufficient evidence to claim that statistical monotonicity holds in this case.

Table 2. R^2 distribution

R^2	Term algorithm	Combined algorithm
0.75-1	52%	57%
0.5-0.74	33%	38%
<0.5	14%	7%

Table 2 summarizes our findings with respect to the R^2 statistics. For each algorithm, we have distinguished between high R^2 value (above 0.75), medium R^2 value (0.5-0.74) and low R^2 value (below 0.5). Low values of R^2 indicate that imprecision level explains less than half of the variance in confidence measures. As an example, consider Fig. 3b, illustrating the linear regression analysis of matching "hotels.com" with "usahotelguid.com(holidayinn)", with $R^2 = 0.44$. In this figure, confidence measures in each imprecision level are scattered rather than being concentrated around the regression line. Therefore, the confidence measures of various imprecision levels are interleaved, and differentiating the various confidence levels becomes much more difficult. As will be discussed in Sect. 6.4, this is an indication of the phenomenon we observe here, where the exact mapping (given as imprecision 0) has a lower confidence measure than permutations from up to $\frac{5}{9}$-imprecision level.

Table 2 shows that in the vast majority of our experiments, both algorithms yielded either medium or high R^2 values. This is an indication of the robustness of the proposed model. In other words, both algorithms generate statistically monotonic mappings. Therefore, the lower the imprecision levels become, the further away a mapping confidence measure would be from the exact mapping.

[7] For large datasets, the normal distribution is assumed. R^2 is an indicator to how "close" the data are to the median at each imprecision level. For normal distributions, the median and the mean (the unbiased estimate of the expected value) are the same. Thus, our experiments validate the statistical monotonicity as presented in Sect. 5.

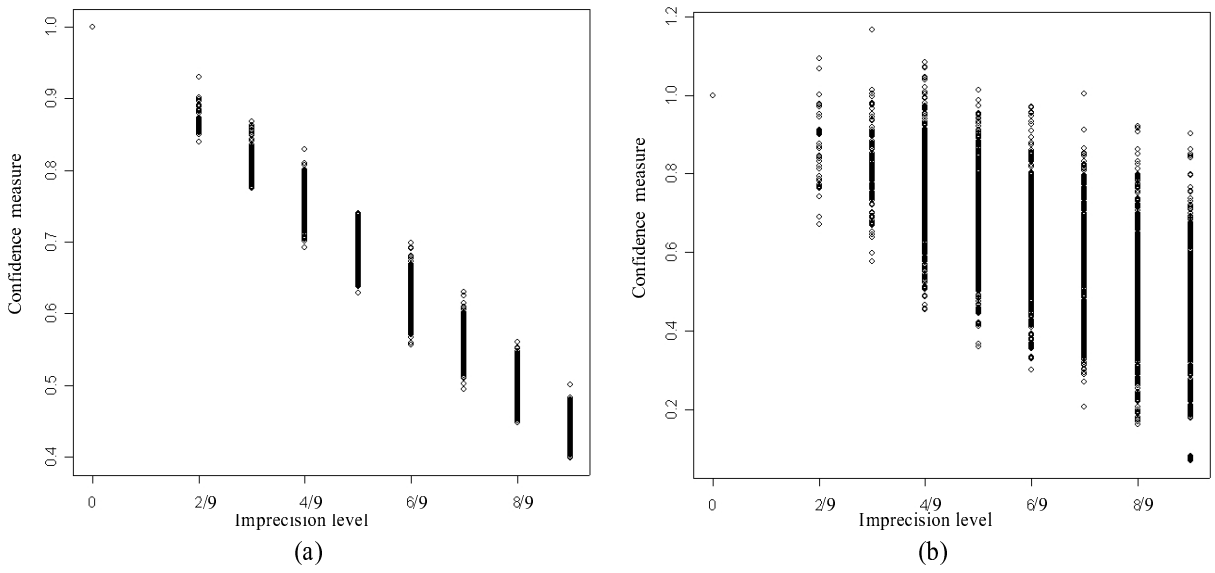

Fig. 3. Linear regression graphs

Table 3. Exact mapping positioning with respect to the best mapping

Rank	Term algorithm	Combined algorithm
0	48%	71%
1-10	29%	19%
11-99	10%	10%
>100	14%	0%
Average rank	105	7

6.3 Pairwise monotonicity

In this section, we look at the relationship between the exact mapping and the best mapping. Clearly, any algorithm that identifies the exact mapping as the best mapping serves its purpose in resolving semantic heterogeneity. Therefore, we first look into the positioning of the exact mapping within an ordered list of all possible mappings. Table 3 summarizes our findings with respect to the positioning of the exact mapping. A rank of 0 means that the algorithm was successful in identifying the exact mapping as the best mapping. Other ranks show the positioning within all possible mappings. We observe that, even if an algorithm fails to identify the exact mapping as the best mapping, high ranking of the exact mapping can assist in identifying it within a small number of trials (see [2] for efficient algorithms that identify top-K mappings).[8] However, if one needs to iterate over all possible permutations, searching the search space becomes intractable. Practically speaking, a good algorithm for automatic semantic reconciliation should take into account the inherent uncertainty of the process. Therefore, it should aim at minimizing the number of iterations required for finding an exact mapping, acknowledging that it is probably impossible to identify an algorithm that would always rank the exact mapping first.

The combined algorithm performs better than the term algorithm. While the combined algorithm manages to identify the exact mapping as the best mapping in 71% of the experiments, the term algorithm manages to achieve the same task in slightly less than half the pairs. An interesting observation is that the combined algorithm performs better when it comes to lower ranking as well. An extreme example involves the mapping of "www.hotels.com" and "HolidayIn.com". In this case, both algorithms have failed to identify the exact mapping as the best mapping. However, while the combined algorithm has ranked the exact mapping second, the term algorithm positioned it in the 235th (!) position. In general, we have observed a "heavy-tail" distribution of the ranking above the exact mapping in the term algorithm, while such a phenomenon was not observed in the case of the combined algorithm. This observation largely explains the average ranking of both algorithms, as provided in Table 3. On average, the combined algorithm positions the exact mapping in the 7th position, while the term algorithm ranks the exact mapping around the 100th position.

6.4 Relationship between monotonicity types

For our next analysis, we shall now present the notion of an ideal mapping, as illustrated in Fig. 4 for $n = 9$. The matrix on the left provides all the pairwise confidence measures of attributes of two schemata. The matrix diagonal represents the exact mapping in which attribute i from one schema is mapped with attribute i from the other schema. The unity matrix represents an ideal (and unrealistic, as observed earlier) scenario, in which a matching algorithm crisply maps attributes of the exact mapping while assigning a 0 value to other attribute combinations. On the right of Fig. 4, we provide the distribution of all mapping permutations according to their confidence

[8] We defer the issue of automatically identifying an exact mapping to a future study. The reader can assume that a human observer is provided with K mappings (either one at a time or as a batch). Alternatively, one can envision a system that utilizes query response in determining the exact mapping. For example, an error message from a Web server can be interpreted as a useless mapping.

	1	2	3	4	5	6	7	8	9
1	1	0	0	0	0	0	0	0	0
2	0	1	0	0	0	0	0	0	0
3	0	0	1	0	0	0	0	0	0
4	0	0	0	1	0	0	0	0	0
5	0	0	0	0	1	0	0	0	0
6	0	0	0	0	0	1	0	0	0
7	0	0	0	0	0	0	1	0	0
8	0	0	0	0	0	0	0	1	0
9	0	0	0	0	0	0	0	0	1

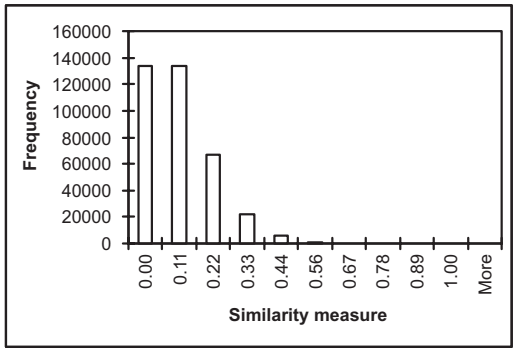

Fig. 4. An ideal mapping scenario

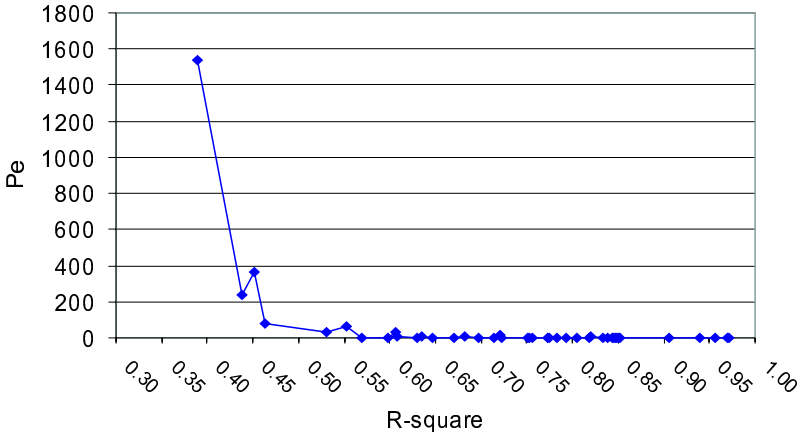

Fig. 5. R^2 vs. P_e

measure. Due to the structure of the matrix, the value of all mappings in a given imprecision level $\frac{i}{n}$ are identical and computed as $\frac{n-i}{n}$. We have designed the graph such that each bar in the graph represents all permutations of a given imprecision level, with smaller imprecision levels on the right and higher imprecision levels on the left.

We are now ready to analyze the relationship between statistical monotonicity and pairwise monotonicity. A priori, one may assume that the latter is indifferent to the behavior of permutations as long as their confidence measure does not exceed that of the exact mapping. In particular, one should not be concerned whether lower imprecision levels demonstrate monotonic behavior. We can therefore hypothesize (just for the sake of argument) that there should be no correlation between statistical monotonicity and pairwise monotonicity. As a measurement of the former we utilize the R^2 statistic. As for the latter, we apply three different measurements, as follows.

- The number of permutations whose confidence measure exceeds that of the exact mapping (P_e). In Table 3 we have summarized the values of P_e as obtained from our experiments.
- The number of permutations whose confidence measure is "close" to that of the exact mapping (P_c). To measure closeness, we look at all permutations whose normalized confidence measure (with respect to the exact mapping) exceeds the confidence measure of the $\frac{1}{9}$-imprecise permutations in the ideal mapping yet do not exceed the confidence measure of the exact mapping.
- The sum of the above measurements, that is, $P_t = P_e + P_c$.

Figure 5 provides the number of permutations whose confidence measure exceeds that of the exact mapping for each experiment as a function of the R^2 value of the regression analysis of the same experiment. There is a negative tendency in the values of P_e as R^2 increases, with few exceptions that can be considered as statistical noise. For example, an experiment with $R^2 = 0.57$ yields $P_e = 0$, and an experiment with $R^2 = 0.97$ yields $P_e = 2$.

Figure 6 provides our analysis with respect to P_c (left) and P_t (right). Here, a strong negative correlation is evident, where for low R^2 values (below 0.7) there is a cluster of permutations around the exact mapping, yet for higher R^2 values the number of permutations with confidence measure close to or above that of the exact mapping declines significantly. We consider this result as a testament to the invalidity of our initial hypothesis in this section. Thus there is a correlation between statistical monotonicity and pairwise monotonicity.

To justify our claim, consider a pictorial illustration of a distribution of confidence measure values according to imprecision levels, given in Fig. 7. This example is a 3D representation of the graphs in Fig. 3. At each level of imprecision, confidence measures seem to be distributed according to the normal distribution with a decreasing mean. As is easily observed, the variance in each imprecision level allows permutations within any given imprecision level to receive high confidence measures. In Fig. 3a, where $R^2 = 0.97$, the small variance does not allow many permutations of low imprecision levels to exceed the confidence level of the exact mapping. In Fig. 3b, we observe a cluster around the best mapping, making the identification of the exact mapping harder to obtain.

Next we analyze the behavior of each of the algorithms separately. Figure 8 provides a side-by-side comparison of P_c as a function of R^2 for the term algorithm (Fig. 8a) and the

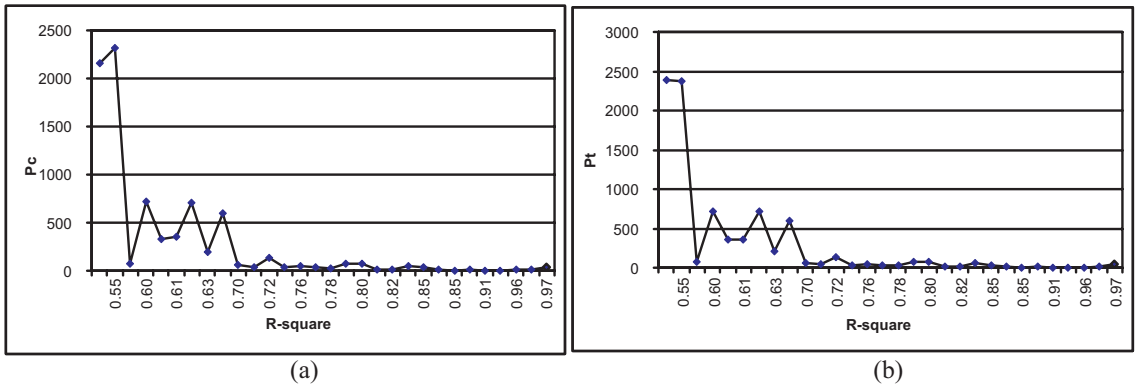

Fig. 6. R^2 vs. P_c/P_t

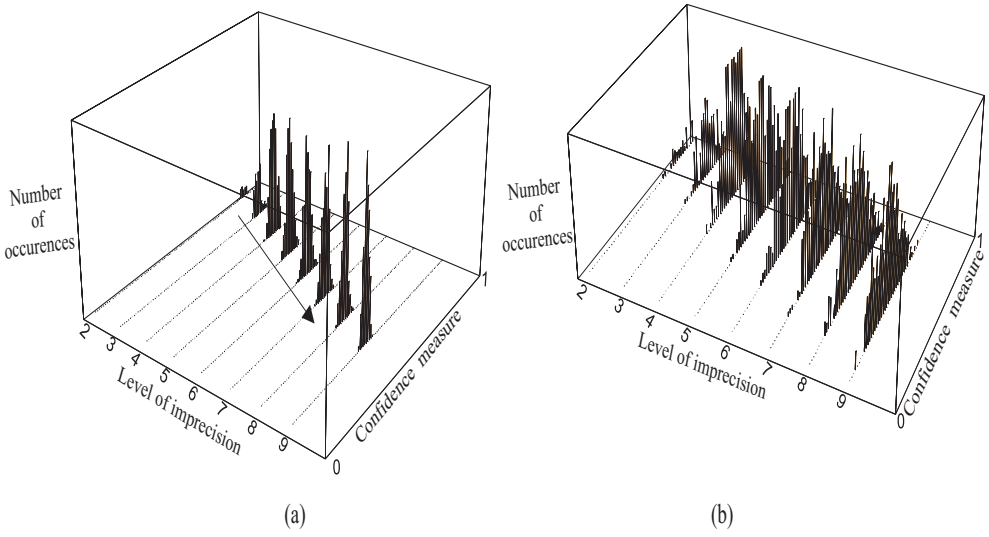

Fig. 7. Similarity measure distribution according to imprecision levels

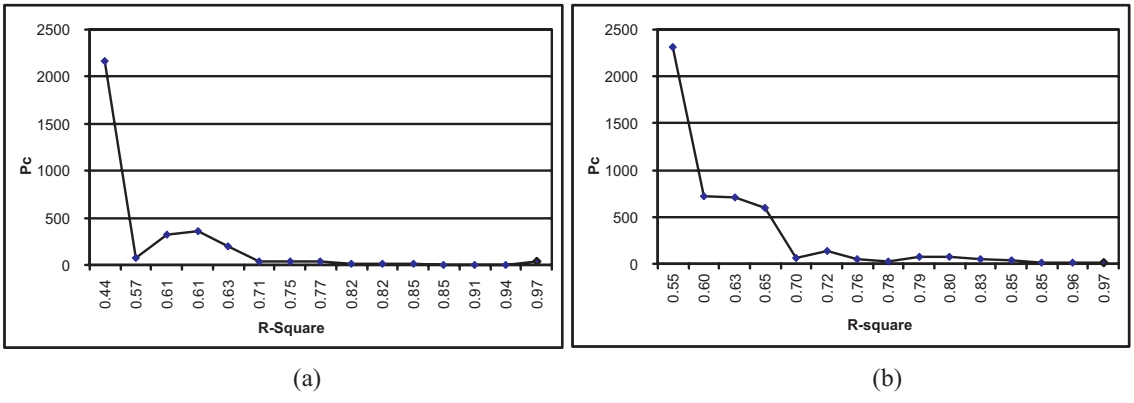

Fig. 8. R^2 vs. P_c: term and combined algorithms

combined algorithm (Fig. 8b). Our conclusion is that the clear negative trend, as observed in Fig. 8b, implies that the combined algorithm yields more predictable results than the term algorithm for any given R^2 value. Along with our analysis from Sect. 6.2, showing that the combined algorithm generates, in general, mappings that are statistically monotonic, one may conclude that the combined algorithm is more likely to rank the exact mapping in a top position among all permutations than the term algorithm.

To conclude the empirical analysis, our main result is that a significant correlation between imprecision level and confidence measure serves as a sufficient evidence for the "goodness" of the algorithm. In particular, such correlation (which we defined through monotonicity) ensures the positioning of the exact mapping sufficiently close to the best mapping. Both algorithms we have experimented with show statistical monotonicity. The combined algorithm, which bases its confidence measure on structural in addition to textual information, seems

to perform better in most cases. The term algorithm maintains statistical monotonicity yet can be improved in many borderline cases, and such functionality is provided by the combined algorithm.

7 Conclusion and future work

We have presented a formal model, capturing the inherent uncertainty in the outcome of automatic schema matching, an analysis of the model properties, and an empirical comparison of applying the proposed framework to two algorithms for filling in a variety of domains in a Web form. The formal model borrows from fuzzy set theory in modeling the uncertainty in the matching process outcome. The theoretical and empirical analyses of the model have yielded the following results:

- For monotonic mappings, one may correlate confidence measure with precision as conceived by a human expert. While monotonicity is a strong notion, weaker notions, such as pairwise monotonicity and statistical monotonicity, suffice for practical purposes (such as identifying the exact mapping within a small number of iterations). Therefore, matching algorithms that generate monotonic mappings (in any form) are well suited for automatic semantic reconciliation.
- Unless attributes in schemata are closely related, mapping confidence cannot utilize any t-norm as its computation vehicle. A preferred operator would come from the fuzzy aggregate operator family, e.g., the $average$ operator. This result provides a theoretical support for the use of variations of the weighted bipartite graph matching for computing schema mapping.
- By comparing two algorithms, namely, the term algorithm and the combined algorithm, we have demonstrated the framework capability to evaluate the suitability of a matching algorithm for automatic semantic reconciliation. In particular, we have shown that both algorithms generate, in general, mappings that are statistically monotonic. However, since the combined algorithm correlates better monotonicity with high ranking of the exact mapping, it is more suitable than the term algorithm for serving in automatic semantic reconciliation scenarios.

The recent steps taken in the direction of automating the schema matching process highlight the critical need for the proposed research. As the automation of the process has already begun to take shape, often without the benefits of thorough research, the study is timely. We envision a multitude of applications of automatic schema matching to the Semantic Web. For example, the research is likely to aid in the design of smart agents that will negotiate over information goods using schema information and provide them with practical tools to combat schema heterogeneity. To this end, we shall conduct a thorough analysis of schema usability to enable us to realistically evaluate the outcomes of a matching algorithm on a practical level. The outcome of the analysis would be the development of robust methods for assessing the usability of mappings to a user. Using these methods, an agent performing on behalf of a user will be able to filter out nonusable mappings so that results to be presented to the user would be of the best quality. We believe that the usability of a mapping can be correlated with its utility to the user. Both involve weighing the utilities of the outcomes and selecting the alternative with the highest expected utility. Therefore, future research will suggest algorithms that will enable such agents to gain a leading edge in the negotiation process by applying economic models to utility functions.

Acknowledgements. The work of Gal was partially supported by the Technion V.P.R. Fund – New York Metropolitan Research Fund, Technion V.P.R. Fund – E. and J. Bishop Research Fund, Fund for the Promotion of Research at Technion, and the IBM Faculty Award for 2003/2004 on "Self-Configuration in Autonomic Computing using Knowledge Management". The work of Montesi is partially supported by the Italian Ministry for Education, Higher Education, and Research (MIUR) as part of the SAHARA project and EU ITEA as part of the ISPI project. Also, the work of Gal and Montesi was partially supported by the Ministry of Science, Culture, and Sport in Israel and by the CNR in Italy. We thank Adi Luboshitz, Ido Peled, and the class of Information Systems and Knowledge Engineering Seminar, fall semester 2002, for their assistance in collecting and analyzing the data.

References

1. Aitchison J, Gilchrist A, Bawden D (1997) Thesaurus construction and use: a practical manual, 3rd edn. Aslib, London
2. Anaby-Tavor A (2003) Enhancing the formal similarity based matching model. Master's thesis, Technion-Israel Institute of Technology, Technion City, Haifa 32000, Israel
3. Aref WG, Barbará D, Johnson S, Mehrotra S (1995) Efficient processing of proximity queries for large databases. In: Yu PS, Chen ALP (eds) Proceedings of the IEEE CS international conference on data engineering, Taipei, Taiwan, 6–10 March 1995. IEEE Press, New York, pp 147–154
4. Arens Y, Knoblock CA, Shen W (1996) Query reformulation for dynamic information integration. In: Wiederhold G (ed) Intelligent integration of information. Kluwer, Dordrecht, pp 11–42
5. Bergamaschi S, Castano S, Vincini M, Beneventano D (2001) Semantic integration of heterogeneous information sources. Data Knowl Eng 36(3): 215–249
6. Berlin J, Motro A (2001) Autoplex: automated discovery of content for virtual databases. In: Batini C, Giunchiglia F, Giorgini P, Mecella M (eds) Proceedings of the 9th international conference on cooperative information systems (CoopIS 2001), Trento, Italy, 5–7 September 2001. Lecture notes in computer science, vol 2172. Springer, Berlin Heidelberg New York, pp 108–122
7. Bernstein PA (2001) Generic model management. In: Batini C, Giunchiglia F, Giorgini P, Mecella M (eds) Proceedings of the 9th international conference on cooperative information systems (CoopIS 2001), Trento, Italy, 5–7 September 2001. Lecture notes in computer science, vol 2172. Springer, Berlin Heidelberg New York, pp 1–6
8. Brodie M (2002) The grand challenge in information technology and the illusion of validity. Keynote lecture at the international federated conference on the move to meaningful Internet systems and ubiquitous computing, Irvine, CA, 30 October–1 November 2002
9. Castano S, de Antonellis V, Fugini MG, Pernici B (1998) Conceptual schema analysis: techniques and applications. ACM Trans Database Sys 23(3):286–332
10. Convent B (1986) Unsolvable problems related to the view integration approach. In: Proceedings of the international conference on database theory (ICDT), Rome, Italy, September 1986.

Also in: Goos G, Hartmanis J (eds) Computer science, vol 243. Springer, Berlin Heidelberg New York, pp 141–156
11. Davis LS, Roussopoulos N (1980) Approximate pattern matching in a pattern database system. Inf Sys 5(2):107–119
12. DeMichiel LG (1989) Performing operations over mismatched domains. In: Proceedings of the IEEE CS international conference on data engineering, Los Angeles, February 1989, pp 36–45
13. DeMichiel LG (1989) Resolving database incompatibility: an approach to performing relational operations over mismatched domains. IEEE Trans Knowl Data Eng 1(4):485–493
14. Doan A, Domingos P, Halevy AY (2001) Reconciling schemas of disparate data sources: A machine-learning approach. In: Aref WG (ed) Proceedings of the ACM-SIGMOD conference on management of data (SIGMOD), Santa Barbara, CA, May 2001. ACM Press, New York
15. Doan A, Madhavan J, Domingos P, Halevy A (2002) Learning to map between ontologies on the semantic web. In: Proceedings of the 11th international conference on the World Wide Web, Honolulu, HI, 7–11 May 2002. ACM Press, New York, pp 662–673
16. Domingos P, Pazzani M (1996) Conditions for the optimality of the simple bayesian classifier. In: Proceedings of the 13th international conference on machine learning, Bari, Italy, 3–6 July 1996, pp 105–112
17. Drakopoulos J (1995) Probabilities, possibilities and fuzzy sets. Int J Fuzzy Sets Sys 75(1):1–15
18. Dwork C, Kumar R, Naor M, Sivakumar D (2001) Rank aggregation methods for the web. In: Proceedings of the 10th international World Wide Web conference (WWW 10), Hong Kong, China, May 2001, pp 613–622
19. Eiter T, Lukasiewicz T, Walter M (2000) Extension of the relational algebra to probabilistic complex values. In: Thalheim B, Schewe KD (eds) Lecture notes in computer science, vol 1762. Springer, Berlin Heidelberg New York, pp 94–115
20. Fagin R (1999) Combining fuzzy information from multiple systems. J Comput Sys Sci 58:83–99
21. Fagin R, Lotem A, Naor M (2001) Optimal aggregation algorithms for middleware. In: Proceedings of the ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems (PODS), Santa Barbara, CA, 21–23 May 2001. ACM Press, New York
22. Fagin R, Wimmers E (1997) Incorporating user preferences in multimedia queries. In: Lecture notes in computer science, vol 1186. Springer, Berlin Heidelberg New York, pp 247–261
23. Fox C (1992) Lexical analysis and stoplists. In: Frakes WB, Baeza-Yates R (eds) Information retrieval: data structures and algorithms. Prentice-Hall, Englewood Cliffs, NJ, pp 102–130
24. Frakes WB, Baeza-Yates R (eds) Information retrieval: data structures and algorithms. Prentice-Hall, Englewood Cliffs, NJ
25. Francis W, Kucera H (eds) Frequency analysis of English usage. Houghton Mifflin, New York
26. Fridman Noy N, Fergerson RW, Musen MA (1937) The knowledge model of prot'eg'e: combining interoperability and flexibility. In: Proceedings of the 12th international conference on knowledge acquisition, modeling and management (EKAW 2000), Juan-les-Pins, France, 2–6 October 2000. Lecture notes in computer science, vol 1937. Springer, Berlin Heidelberg New York, pp 17–32
27. Fridman Noy N, Musen MA (1999) Smart: automated support for ontology merging and alignment. In: Proceedings of the 12th Banff workshop on knowledge acquisition, modeling and management, Banff, Alberta, Canada, 16–21 October 1999
28. Fridman Noy N, Musen MA (2000) PROMPT: algorithm and tool for automated ontology merging and alignment. In: Proceedings of the 17th national conference on artificial intelligence (AAAI-2000), Austin, TX, 30 July–3 August 2000, pp 450–455
29. Gal A (1999) Semantic interoperability in information services: experiencing with CoopWARE. SIGMOD Rec 28(1):68–75
30. Gal A, Modica G, Jamil HM (2003) Automatic ontology matching using application semantics. Submitted for publication. Available upon request from avigal@ie.technion.ac.il
31. Galil Z (1986) Efficient algorithms for finding maximum matching in graphs. ACM Comput Surv 18(1):23–38
32. Gonzales RC, Thomanson MG (1978) Syntactic pattern recognition – an introduction. Addison-Wesley, Reading, MA
33. Hajek P (1998) The metamathematics of fuzzy logic. Kluwer, Dordrecht
34. Hull R (1997) Managing semantic heterogeneity in databases: A theoretical perspective. In: Proceedings of the ACM SIGACT-SIGMOD-SIGART symposium on principles of database systems (PODS), Tucson, AZ, 13–15 May 1997. ACM Press, New York, pp 51–61
35. Jarrar M, Meersman R (2002) Formal ontology engineering in the DOGMA approach. In: Proceedings of the international federated conference on the move to meaningful Internet systems and ubiquitous computing, Irvine, CA, October 2002, pp 1238–1254
36. Kahng J, McLeod D (1996) Dynamic classification ontologies for discovery in cooperative federated databases. In: Proceedings of the 1st IFCIS international conference on cooperative information systems (CoopIS'96), Brussels, Belgium, June 1996, pp 26–35
37. Klement EP, Mesiar R, Pap E (2000) Triangular norms. Kluwer, Dordrecht
38. Klir GJ, Yuan B (eds) Fuzzy sets and fuzzy logic. Prentice-Hall, Englewood Cliffs, NJ
39. Lakshmanan LVS, Leone N, Ross R, Subrahmanian VS (1997) Probview: A flexible probabilistic database system. ACM Trans Database Sys (TODS) 22(3):419–469
40. Langley P, Iba W, Thompson K (1992) An analysis of bayesian classifiers. In: Proceedings of the 10th national conference on artificial intelligence, San Jose, CA, 12–16 July 1992, pp 223–228
41. Levenstein IV (1966) Binary codes capable of correcting deletions, insertions, and revrsals. Cybern Control Theory 10(8):707–710
42. Madhavan J, Bernstein PA, Domingos P, Halevy AY (2002) Representing and reasoning about mappings between domain models. In: Proceedings of the 18th national conference on artificial intelligence and the 14th conference on innovative applications of artificial intelligence (AAAI/IAAI), Edmonton, Alberta, Canada, 28 July–1 August 2002, pp 80–86
43. Madhavan J, Bernstein PA, Rahm E (2001) Generic schema matching with Cupid. In: Proceedings of the international conference on very large data bases (VLDB), Rome, Italy, September 2001, pp 49–58
44. Maedche A, Staab S (2002) Measuring similarity between ontologies. In: Proceedings of the 13th international conference on knowledge engineering and knowledge management: ontologies and the semantic Web (EKAW 2002), Siguenza, Spain, October 2002, pp 251–263
45. McGuinness DL, Fikes R, Rice J, Wilder S (2000) An environment for merging and testing large ontologies. In: Proceedings of the 7th international conference on principles of knowledge representation and reasoning (KR2000), Breckenridge, CO, 11–15 April 2000, pp 483-493
46. Mena E, Kashayap V, Illarramendi A, Sheth A (2000) Imprecise answers in distributed environments: Estimation of information

loss for multi-ontological based query processing. Int J Coop Inf Sys 9(4):403–425
47. Miller RJ, Haas LM, Hernández MA (2000) Schema mapping as query discovery. In: El Abbadi A, Brodie ML, Chakravarthy S, Dayal U, Kamel N, Schlageter G, Whang K-Y (eds) Proceedings of the international conference on very large data bases (VLDB), Cairo, Egypt, 10–14 September 2000. Morgan Kaufmann, San Francisco, pp 77–88
48. Miller RJ, Hernàndez MA, Haas LM, Yan L-L, Ho CTH, Fagin R, Popa L (2001) The Clio project: managing heterogeneity. SIGMOD Rec 30(1):78–83
49. Modica G, Gal A, Jamil H (2001) The use of machine-generated ontologies in dynamic information seeking. In: Batini C, Giunchiglia F, Giorgini P, Mecella M (eds) In: Proceedings of the 9th international conference on cooperative information systems (CoopIS 2001), Trento, Italy, 5–7 September 2001. Lecture notes in computer science, vol 2172. Springer, Berlin Heidelberg New York, pp 433–448
50. Moulton A, Madnick SE, Siegel M (1998) Context mediation on Wall Street. In: Proceedings of the 3rd IFCIS international conference on cooperative information systems (CoopIS'98), New York, August 1998. IEEE-CS Press, New York, pp 271–279
51. Nadler M, Smith E (1993) Pattern recognition engineering. Wiley, New York
52. Nestorov S, Abiteboul S, Motwani R (1998) Extracting schema from semistructured data. In: Haas LM, Tiwary A (eds) Proceedings of the ACM-SIGMOD conference on management of data (SIGMOD), Seattle, June 1998. ACM Press, New York, pp 295–306
53. Omelayenko B (2002) RDFT: a mapping meta-ontology for business integration. In: Proceedings of the workshop on knowledge transformation for the semantic Web (KTSW 2002) at the 15th European conference on artificial intelligence, Lyon, France, July 2002, pp 76–83
54. Ouksel AM, Naiman CF (1994) Coordinating context building in heterogeneous information systems. J Intell Inf Sys 3(2):151–183
55. Palopoli L, Terracina LG, Ursino D (2000) The system DIKE: towards the semi-automatic synthesis of cooperative information systems and data warehouses. In: Proceedings of current issues in databases and information systems, East European conference on advances in databases and information systems. Held jointly with the international conference on database systems for advanced applications (ADBIS-DASFAA 2000), Prague, Czech Republic, 5–8 September 2000, pp 108–117
56. Rahm E, Bernstein PA (2001) A survey of approaches to automatic schema matching. J Very Large Data Bases 10(4):334–350
57. Schalkoff R (1992) Pattern recognition: statistical, structural, and neural approaches. Wiley, New York
58. Schuyler PL, Hole WT, Tuttle MS (1993) The UMLS (Unified Medical Language System) metathesaurus: representing different views of biomedical concepts. Bull Med Libr Assoc 81:217–222
59. Sheth A, Larson J (1990) Federated database systems for managing distributed, heterogeneous, and autonomous databases. ACM Comput Surv 22(3):183–236
60. Sheth A, Rusinkiewicz M (1993) On transactional workflows. Data Eng Bull 16(2):37–40
61. Soergel D (1985) Organizing information: principles of data base and retrieval systems. Academic, Orlando
62. Spyns P, Meersman R, Jarrar M (2002) Data modelling versus ontology engineering. ACM SIGMOD Rec 31(4):12–17
63. Valtchev P, Euzenat J (1997) Dissimilarity measure for collections of objects and values. In: Liu X, Cohen PR, Berthold MR (eds) Proceedings of the 2nd international symposium on advances in intelligent data analysis, reasoning about data (IDA-97), London, 4–6 August 1997. Lecture notes in computer science, vol 1280. Springer, Berlin Heidelberg New York, pp 259–272
64. Van Harmelen F, Fensel D (1999) Practical knowledge representation for the web. In: Proceedings of the IJCAI-99 workshop on intelligent information integration, in conjunction with the 16th international joint conference on artificial intelligence, Stockholm, Sweden, 31 July 1999. Proceedings of the CEUR workshop, Stockholm, Sweden, 31 July 1999, vol 23
65. Vickery BC (1966) Faceted classification schemes. Graduate School of Library Service, Rutgers State University, New Brunswick, NJ