Object Identification Quality

Mattis Neiling*,[1], Steffen Jurk*,[2], Hans-J. Lenz[1] and Felix Naumann[3]

[1] FU Berlin, Institute for Information Systems
{mneiling|hjlenz}@wiwiss.fu-berlin.de

[2] BTU Cottbus, Dept. of Database and Information Systems
sj@informatik.tu-cottbus.de

[3] IBM Almaden Research Center
felix@almaden.ibm.com

***Abstract*— Research and industry has tackled the object identification problem of data integration in many different ways. This paper presents a framework, that allows the evaluation of competing approaches. To this end, complexity measures and data characteristics are introduced, which reflect the hardness of a given object identification problem. All characteristics can be estimated by use of simple SQL queries and simple calculations. Following the principle of benchmark definitions we specify a *test framework*. It consists of a test database and its characteristics, quality criteria, and a test specification. Adequate measures needed for the *correctness* criterion of the benchmark are given.**

A running example of the *Berlin Online Apartment-Advertisements database* (BOA) illustrates the approach. The BOA-database is freely available at `www.wiwiss.fu-berlin.de/lenz/boa/`.

I. Motivation

Even though quality cannot be defined,
you know what it is.
Robert Pirsig

For databases, object identification is the task of finding multiple database records, which represent the same real world object, in particular when no global identifier is available. Object identification becomes essential, when data about the same real-world objects is distributed over two or more data sources. Different methods and software packages tackle this problem: Known methods are *Record Linkage* [1], [2], [3], the *Nearest-Neighbor-* and *k-way Sorting-Method* [4], [5], *Data Lineage Tracing* [6] or the generic approach *Identification-by-means-of-Classification* introduced by two of the authors, c.f. [7], [8]. Commercial software packages are INTEGRITY™ [9], or the MERGE/PURGE component of the CENTRUS™ data quality software [10] for example. A wide range of software dealing with the standardization and de-duplication of address data exists, e.g. FUZZY! POST™ [11]. In this paper we provide a *test framework* to enable the comparative analysis of such competing approaches.

Currently, no publicly available test data exist — researchers and software vendors usually evaluate their solutions on domain-specific data. Unfortunately, data used for testing typically is confident, such as customer data or census data. Thus, a publicly available collection of test data for independent evaluation is required. In this paper we discuss properties, that such test databases should fulfill and recommend quality criteria and a test specification.

*Part of this work was supported by the Berlin-Brandenburg Graduate School in Distributed Information Systems (DFG grant no. GRK 316)

This paper is organized as follows: First, we present the general object identification problem and the structure and properties of its corresponding object identification solution. Next, we define a *test database* and discuss *characteristics* of object identification problems, which determine their *hardness*. Then, we specify a *test framework* for object identification, consisting of a test database, its characteristics, quality criteria, and a test specification. Finally, we discuss the details of quality criteria and the specification.

II. The Object Identification Problem

Looking for duplicated records
within a large real-world database
is like fishing in troubled waters
User experience

Object identification on a database A, providing data on an universe of real-world objects like people, books, etc., describes the following problem:

$$\text{"Which database records } a, b \in A \text{ refer to the same real-world object?"}, \quad (1)$$

or, more comprehensive, "Which records $a, b \in A$ are duplicates?"

We are interested in a solution of (1) concerning additional constraints, implied by background information (metadata) on A. For example, apartments of a different size (square meters) do definitely not refer to the same real-world object, whereas appartments with a different rent might still refer to the same real-world object. In this work metadata are characterized as a set of constraints $\mathcal{C}$ on A. We denote any particular solution of (1) as an object identification solution, e.g., an algorithm that, given A and $\mathcal{C}$, returns the pairs assessed as duplicates in A. Formally, a solution can be described as a *decision rule* or *classifier* $\delta_{\mathcal{C}} : A \times A \rightarrow \{0, 1\}$, with

$$\delta_{\mathcal{C}}(a, b) = 0 \iff a \text{ and } b \text{ are classified as duplicates.} \quad (2)$$

Remark II.1. The decision $\delta_{\mathcal{C}}(a, b) = 1$ indicates, that a and b could not be classified as duplicates—the hypothesis *a and b are duplicates* would be rejected. From test theory we know that two kinds of errors occur, namely the rejection

of matches (so-called α-error), and the acceptance of non-matches as duplicates (so-called β-error), c.f. section IV-B. Lim et al. [12] propose to distinguish three decisions: *identical/undetermined/non matching*.[1] However, a third *undetermined* decision causes the difficulty of correct error estimation. If the number of undetermined cases increases, both the α- and β-error tend to decrease. Therefore a limit $\varepsilon \in [0,1)$ for the number of undetermined cases has to be set beforehand, such that $|\{\text{undetermined pairs } (a,b) \in A \times A\}| \leq \varepsilon \cdot |A|$. Typical sizes for the limit ε are 0, 0.1, 0.01, etc.

In the next section we introduce and discuss the hardness of an object identification problems. The hardness is related to the quality of solutions. The complexity and difficulty of a problem influence the quality: Even if a solution is regarded as good for a specific class of problems, it might be outperformed by other solutions for a different class of problems. This fact is similar to the NO FREE LUNCH situation for *optimization* and *supervised learning* problems (c.f. details [13]) and it applies also for the object identification problem.

III. The Hardness of Object Identification Problems

Due to the uniqueness of every object identification problem, it is rather difficult to define its hardness. Before we define the *hardness* we restrict the database A to a single relation, e.g., A is made of a single table with the n attributes $X = (X_1, \ldots, X_n)$. This assumption is reasonable, since the identifying information of real-world objects belonging to *one* universe can be made uniformly and flat in the most cases. If a value of an attribute X_i is not provided for the j-th tuple $a_j \in A$, we set $X_i(a_j) = \text{NULL}$.

The indicator *hardness* of an object identification problem for an database A depends (1) on given metadata for A, it is a set of *semantic constraints* $\mathcal{C}$, (2) on the number of pairs, which can be built from A, it is $|A|^2/2$, and (3) on the selectivity of the attribute set $Y \subset X$, which contains identifying information. The hardness measure reflects the I/O complexity to access pairs of tuples from A.

Definition III.1 (Hardness). Let $Y \subset X$ be an attribute set and let $\phi_{log_{10}} : \mathbb{R}_{\geq 0} \to [0,1),\ v \mapsto \phi_{\log_{10}}(v) := \left(1 - \frac{1}{1+\log_{10}(1+v)}\right)$ be a function.[2] Then the *hardness* of the object identification problem for A is defined by

$$\text{hardness}(Y, A, \mathcal{C}) := \phi_{log_{10}}\left(\Psi(\mathcal{C}) \cdot \tfrac{|A|^2}{2} \cdot \Psi_{sel}\right), \quad (3)$$
$$\text{with } \Psi_{sel} := 1 + \theta_0 - \max_{Y' \subset Y}\left(\text{selectivity}(Y', A)\right),$$

where $\Psi(\mathcal{C}) \in [0,1]$ is a sophisticated weight function (c.f. Section III-B, Definiton III.16), dependent from metadata, namely from a set of *characteristics* $\mathcal{C}$ supplied with an object identification problem. $\theta_0 \in (0,1)$ is chosen, such that the third factor in (3) is greater zero, e.g. $\theta_0 = \frac{1}{100}$.

[1] The *undetermined* cases require a manual inspection afterwards.

[2] $\phi_{\log_{10}}$ is a bijection with $\phi^{-1}_{\log_{10}} : [0,1) \to \mathbb{R}_{\geq 0},\ \xi \mapsto \left(10^{\frac{\xi}{1-\xi}} - 1\right)$.

Remark III.2. The suggested indicator for the *hardness* measure (3) is defined as an *product*, because we suppose, that the *compensability axiom* holds: The enlargement of one factor can be compensated with the diminution of another factor, e.g., the enlargement of the size $|A|$ might be compensated by improved metadata $\mathcal{C}$ for $|A|$, such that the $\Psi(\mathcal{C})$-value is decreased.

Each *characteristic* C_k is a numerical function defined on a subset of attributes X_i and subsets of A. Practioners might determine their knowledge by a set of *semantic constraints* $\mathcal{C}$. Each constraint in $\mathcal{C}$ characterizes metadata for a given data set A and is termed as $[C_k \diamond v],\ \diamond \in \{<, \leq, =, \geq, >, \approx\}$ and v as a nonnegative numeric value.

An object identification solution might use the given *semantic constraints* to improve the precision of its result. Practioners can fix *semantic constraints* for a given problem A. The constraints can be evaluated precisely, if a correct list of duplicates is available.

Definition III.3. $\mathcal{D} = (A, \mathcal{C}, \mathsf{Same})$ is a **Test Database** (for object identification), if

- A is a database containing tuples $a \in A$ from the same universe of real-world objects,
- $\mathcal{C}$ is a finite set of semantic constraints describing characteristics of A (metadata), and
- $\mathsf{Same} \subset (A \times A)$ is an equivalence relation and contains the pairs which are equivalent *in reality*. The Same-relation is complete if (1) for each pair $(a,b) \in \mathsf{Same}$, a and b really provide data referring to equivalent real-world objects, written $a \equiv b$ and (2) for each pair $(a,b) \in (A \times A) \backslash \{\mathsf{Same}\}$, a and b refer to nonequivalent real-world objects, written $a \not\equiv b$.

Remark III.4. There might exist a difference between the *equivalence* and *identity* of two real-world objects. The *equivalence* of real-world objects must be determined by a domain specific concept of identity. For example, the abstract concept of identity for a library catalogue can be determined by editions of books. Note, that an object identification solution δ_C might be imprecise and therefore different from Same, it is $\mathsf{Same} \neq \{(a,b) \mid \delta_{\mathcal{C}}(a,b) = 0\}$.

We can precisely evaluate each function C_k on Same. The resulting value is denoted as C_k^{S} and the set of this constraints is denoted as $\mathcal{C}^{\mathsf{S}}$. We assume that $\mathcal{C}$ and $\mathcal{C}^{\mathsf{S}}$ are consistent. That is, for each constraint $C_k \diamond v$, $C_k^{\mathsf{S}} \diamond v$ holds true.

Example III.5 *(BOA — Berlin Online Apartment Ads).* Database A contains advertisements of Berlin apartments. These ads were extracted from four online editions of the Berlin newspapers *Tagesspiegel* and *Berliner Morgenpost*[3] at the 18th and 25th of May 2002, respectively. From *Tagesspiegel* we extracted 1,507 and 1,643 ads (forming the subsets $A_1, A_2 \subset A$), while *Berliner Morgenpost* supplied 2,962 and 3,730 ads of apartments in Berlin (forming $A_3, A_4 \subset A$). The database comprises a single table with

[3] `www.tagesspiegel.de` and `www.mopo-immo.de`.

approximately 10,000 rows. Common attributes in all ads are *FullText*, *DistrictOfBerlin*, *Rent*, *Rooms*, and *Size*. If available, we extracted *Phone* (for 96%), *Street* (for 66%), *Floor* (for 43%) and other attributes.

To simplify data processing we performed the following steps: (1) We annotated each record with an attribute identifying its lineage, (2) We annotate each record with an attribute *ID* as unique key, (3) We sort the tuples of A in increasing order w.r.t. *ID*, $A = (a_1, \ldots, a_N)$, such that holds

$$\forall\, a_i, a_j \in A:\ i < j \Longrightarrow \mathit{ID}(a_i) < \mathit{ID}(a_j),$$

we notate the ordering on A by $a_i < a_j$ for $i < j$, (4) To avoid redundancy, we report only $|A'|-1$ pairs in the *Same*-relation instead of all $\binom{|A'|}{2}$ possible pairs for a set of duplicates $A' \subset A$ — t.i. all tuples $a \in A'$ refer to the same real-world object. Therefore, let $I \subset \{1, \ldots, |A'|\}$ be an index set, such that $A' = \{a_i\}_{i \in I}$. We set $i_0 := \min(i \in I)$. Then we report for each duplicate a_i, $i \in I, i \neq i_0$ *exactly one* pair in the *Same*-relation, namely (a_{i_0}, a_i). Nevertheless, all pairs of duplicates can be derived from the *Same*-relation by

$$a \equiv b \Longleftrightarrow \big((a = b) \vee (a, b) \in \mathit{Same} \vee (b, a) \in \mathit{Same} \vee \vee\, (\exists a_0 \in A : (a_0, a) \in \mathit{Same} \wedge (a_0, b) \in \mathit{Same})\big).$$

A. Complexity Measures and Characteristics

In the following subsection we proceed to define a set of complexity measures and a set of characteristics. Measures are metadata that can be derived automatically, while characteristics can only be determined by an expert. Both sets of properties will then be used to evaluate the difficulty of finding duplicates in a database — the *hardness* of the object identification problem. Most properties are accompanied by our running example and the results are shown in Figure 3.

- Set of complexity measures: (1) The size[4] $|A|$ as the number of tuples in A, (2) the selectivity(Y, A) of an attribute set Y of A, and (3) the fraction of NULL-values for an attribute set Y in A: null(Y, A).
- Set of characteristics: (1) Existence of *semantic keys* and *anti-keys* $Y \subset X$ and the reduction_rate(A) of pairs from $A \times A$ achieved by these keys, (2) the share of duplicates in A: duplicates(A), (3) the *goodness of identification* of attribute sets: goodness(Y), and (4) the accuracy(Y) of attribute sets.

1) Selectivity and Missing Values: A necessary condition for object identification with attribute sets is high selectivity. The selectivity for an attribute set $Y \subset X$ is defined as:

$$\begin{aligned}\text{selectivity}(Y, A') &:= \frac{|A'/\mathrm{dom}\, Y|}{|A'|} \qquad (4)\\ &= \frac{|\{\,\{a \in A' \mid Y(a) = y\} \mid y \in \mathrm{dom}\, Y\}|}{|A'|},\end{aligned}$$

where $A'/\mathrm{dom}\, Y$ is induced by the equivalence relation $(A, =_Y)$. Obviously, for any candidate key we get [selectivity$(Y, A) = 1$]. But the converse is not true — high selectivity is no indicator for the key property, since $[a \equiv b \Longrightarrow Y(a) = Y(b)]$ must not hold in general. Missing values influence the identification. We can calculate the occurrence of missing values for attribute sets $Y \subset X$ by

$$\text{null}(Y, A') := \frac{|\{a \in A' \mid \exists Y_i \in Y : Y_i(a) = \text{NULL}\}|}{|A'|}. \quad (5)$$

Claim III.6. *For all $Y \subset X$ and for all $A' \subset A$ holds* null$(Y, A') \leq 1 -$ selectivity(Y, A').

2) Semantic Keys and Anti-Keys: Knowledge about keys can greatly reduce the number of pairs of $(A \times A)$ to be checked.

Definition III.7. Y is an *semantic key* on $A' \subseteq A$, if Y fulfills the *key property*

$$[\forall a, b \in A' : Y(a) = Y(b) \Longleftrightarrow a \equiv b]. \quad (6)$$

If a *semantic key* Y exists for a subset A', object identification can be performed on A' — using Y as an identifier. Then, the number of pairs in $A \times A$ decreases by $\frac{1}{2}(|A'|-1)(|A'|-2) \approx \frac{1}{2}|A'|^2$.

Example III.8 *(Semantic Key).* Let A be a library catalogue, let $A^{(i)} \subset A$ and let $Y^{(1)} = (\mathit{ISBN})$ and $Y^{(2)} = (\{\mathit{Author}_1, \mathit{Author}_2, \ldots\}, \mathit{Title}, \mathit{Edition}, \mathit{Publisher})$ attribute sets. If [null$(Y^{(i)}, A^{(i)}) = 0$] for $A^{(i)} \subset A$ then $Y^{(i)}$ forms a semantic key, [semKey$(Y^{(i)}, A^{(i)}) = 1$].

Definition III.9. Let Y be an attribute set, dist : $(\mathrm{dom}\, Y)^2 \to \mathbb{R}_{\geq 0}$ a distance measure, and $\Delta > 0$. Y is an *semantic anti-key* on $A' \subseteq A$, if Y indicates non-equalness of real-world objects, if the distance measure exceeds Δ, t.i. $\forall a, b \in A'$: dist$\big(Y(a), Y(b)\big) \geq \Delta \Longrightarrow a \not\equiv b$.

Anti-keys form k pairwise disjunct duplicate-free subsets $A^{(i)} \subset A$ ($i = 1, \ldots, k$; $k > 1$), such that the number of ordered pairs in $A \times A$ reduces to $\frac{1}{2}\sum_{i=1}^{k}(|A^{(i)}|-1)(|A^{(i)}|-2) \approx \frac{1}{2}\sum_{i=1}^{k}|A^{(i)}|^2$. The value of k depends on the *selectivity* of Y in A. Anti-keys are a very efficient way for reduction, since the reduction rate grows exponential w.r.t. the number of applied anti-keys in the best case. Finally, [reduction_rate$(A) = c$] denotes the ratio of all pairs removed by such keys.

Example III.10 *(BOA cont.).* Our data has been extracted from four sources $A_i \subset A$, $1 \leq i \leq 4$ (online editions). Each source provides a semantic key [semKey(Y, A_i)] for some attribute Y indicating that each source is free of duplicates. Thus the overall number of pairs $48,417,720$ of potential duplicates is reduced to $34,604,058$. Further the following anti-keys are given: [antiKey(*DistrictOfBerlin*, $A, 0$)], [antiKey(*Rooms*, $A, 0.5$)], and [antiKey(*Size*, $A, 1.0$)]. The number of ordered pairs decreases to 51,593.[5] After all, a reduction rate of 99.8761% has been achieved.

[4]Size is a relative measure, since hardware and computational power increases over time.

[5]This number might be increase approximately by 10%, if records with exceptional or implausible values must be compared with other records.

3) The Number of Duplicates: Sometimes metadata contain information about the expected number of duplicates in subsets of A.

Let $A' \subset A$. Then the characteristic duplicates is defined as follows:

$$\text{duplicates}(A') := \left|\{a \in A' \mid \exists a_0 \in A' \colon (a > a_0 \wedge a \equiv a_0\}\right|. \tag{7}$$

Note, that the expression $\text{duplicates}(A') = 0$ indicates duplicate-freeness of A', while $\text{duplicates}(A') \leq |A'| - 1$ specifies, that the number of duplicates in A' is unknown.

Let $A', A'' \subset A$. The *overlap* of the real-world objects, *both* A' and A'' refer to, is given by

$$\begin{aligned}\text{overlap}(A', A'') := \ &\text{duplicates}(A' \cup A'') - \\ &- \text{duplicates}(A') - \text{duplicates}(A'')\end{aligned} \tag{8}$$

Let $\{A^{(i)}, i = 1, \ldots, k\}$ be a partitioning of A. Assume the constraints $[\text{overlap}(A^{(i)}, A^{(j)}) \diamond c_{ij}]$, $[\text{duplicates}(A^{(i)}) \diamond c_i]$,with $1 \leq i \leq j \leq k$, an operator $\diamond \in \{<, \leq, =, \geq, >, \approx\}$ and $c_i, c_{ij} \in \mathbb{N}$. Applying (8), these expressions can be aggregated to the constraint

$$\left[\text{duplicates}(A) \diamond \sum_{i=1}^{k} \left(c_i + \sum_{j=i+1}^{k} c_{ij}\right)\right]. \tag{9}$$

It follows, that $c_{\min}, c_{\max} \in \mathbb{N}$ with $0 \leq c_{\min} \leq c_{\max} \leq |A| - 1$ exist, such that

$$[c_{\min} \leq \text{duplicates}(A) \leq c_{\max}]. \tag{10}$$

Example III.11 *(BOA cont.).* For the sources $A_1, \ldots, A_4$ we have the following information: Between 5% and 30% of the apartments are announced in two successive weekend-editions of the *Tagespiegel* and the *Berliner Morgenpost* and the overlap of both newspapers of the same weekend is between 10% and 20%. If we consider the duplicate-freeness of each online edition we get by use of (9) $[537 < \text{duplicates}(A) < 2,636]$. Of course we still do not know, which records in A are in fact duplicates. Satisfyingly, through a manual search we have found 2,187 duplicates — $[duplicates^{\mathsf{S}}(A) = 2,187]$. But for the successive weekend-editions of the *Berliner Morgenpost* we found above 50% duplicates. We conclude from this result, that for practical applications it is rather difficult to determine the number of duplicates.

4) Goodness of Identification: In section III-A.2 we introduced the concept of keys. However, for a given attribute Y we are also interested in its capability to fulfill the key property. The *goodness of identification* of an attribute set $Y \subset X$ can be analyzed with the accuracy of Y, i.e., the probability of $Y(a) = Y(b)$ for matched pairs (a, b). Since the equality of the attribute values of Y is only of interest for identification, if $Y(a) = Y(b)$ occurs infrequently for non-matched pairs, we apply the likelihood ratio $\lambda \in \mathbb{R}_{\geq 0}$,

$$\lambda(Y) := \frac{P\big(Y(a) = Y(b) \mid a \equiv b\big)}{P\big(Y(a) = Y(b) \mid a \not\equiv b\big)}. \tag{11}$$

Resolving the conditional probabilities in (11) leads to[6]

$$\begin{aligned}\lambda(Y) &= \\ &= \frac{P\big(Y(a) = Y(b),\, a \equiv b\big)\, P\big(a \not\equiv b\big)\, P\big(Y(a) = Y(b)\big)}{P\big(Y(a) = Y(b),\, a \not\equiv b\big)\, P\big(a \equiv b\big)\, P\big(Y(a) = Y(b)\big)} \\ &= \frac{P\big(a \equiv b \mid Y(a) = Y(b)\big)}{P\big(a \not\equiv b \mid Y(a) = Y(b)\big)} \cdot \underbrace{\frac{P\big(a \not\equiv b\big)}{P\big(a \equiv b\big)}}_{const.}.\end{aligned} \tag{12}$$

Hence, the likelihood ratio (11) is influenced by the ratio of the apriori probabalities of unmatched and matched pairs in A. Thus, we define the rescaled likelihood ratio $\lambda^*(Y)$ by

$$\lambda^*(Y) := \lambda(Y) \cdot \frac{P\big(a \equiv b\big)}{P\big(a \not\equiv b\big)}. \tag{13}$$

If (13) is transformed by the bijection $\phi : \mathbb{R}_{\geq 0} \cup \{\infty\} \to [0, 1]$, $\xi \mapsto \big(1 - \frac{1}{1+\xi}\big)$, $\phi(\infty) = 1$, the goodness of Y for identification can be measured by the *confidence value* $\text{goodness}(Y) \in [0, 1]$ on a test database $\mathcal{D} = (A, \mathcal{C}, \mathit{Same})$ as follows:

$$\text{goodness}(Y) := \phi\left(\hat{\lambda}^*(Y)\right) = \left(1 - \frac{1}{1 + \hat{\lambda}^*(Y)}\right) \tag{14}$$

By use of (12) and (13) we simplify (14) as follows

$$\begin{aligned}&\text{goodness}(Y) = \\ &= \frac{\hat{P}\big(Y(a) = Y(b) \mid a \equiv b\big)\, \hat{P}\big(a \equiv b\big)}{\sum_{\diamond \in \{\equiv, \not\equiv\}} \hat{P}\big(Y(a) = Y(b) \mid a \diamond b\big)\, \hat{P}\big(a \diamond b\big)}\end{aligned} \tag{15}$$

We set $[\text{goodness}(Y) = 0]$ if the denominator in (15) vanishes. The estimates of the conditional probabilities in (15) are given by (with $\diamond \in \{\equiv, \not\equiv\}$)

$$\hat{P}\big(Y(a) = Y(b) \mid a \diamond b\big) := \frac{\left|\{(a, b) \mid Y(a) = Y(b) \wedge a \diamond b\}\right|}{\left|\{(a, b) \mid a \diamond b\}\right|}.$$

As an extension of semantic keys we introduce the concept of *approximate keys*.

Definition III.12. Y is a *approximate key* for A with confidence factor $p > \frac{1}{2}$, **iff** $\text{goodness}(Y) = p$.

The higher the confidence factor, the more correct the identification can be carried out by Y. But in contrast to semantic keys, the key property (6) holds for an approximate key only with its confidence p, e.g., for $[\text{goodness}(Y) \approx \frac{1}{2}]$ we have $\hat{\lambda}(Y) \approx 1$, that is the *odds* are 1:1. Thus, if the $\text{goodness}(Y)$ is below $\frac{1}{2}$, the use of Y as an approximate key becomes unreasonable.

Further, in analogy to anti-keys, a notion of anti-goodness can be defined. Similar to semantic anti-keys, approximate anti-keys with a high confidence can be used to reduce the number of pairs left for comparison. We omit details.

[6]The conditional probability is defined for events A, B by

$$P(A \mid B) := \frac{P(A,\, B)}{P(B)}.$$

Example III.13 *(BOA cont.).* For some attributes of the BOA database the goodness values are given in Figure I. Note that the attribute *FullText* has a high goodness, $[\text{goodness}^{\mathsf{S}}(\textit{FullText}) = 0.9887]$ and no occurrences of null values $[\text{nulls}^{\mathsf{S}}(\textit{FullText}) = 0]$ and consequently $[\text{approxKey}(\textit{FullText}, A)]$.

5) Accuracy: Errors have a deep impact on data quality. In case of quantitative data recorded from observations, errors can often be modelled by Gaussian noise. Then outliers can be detected with high probability. Unfortunately, errors in real-world data are often more complicated: Many attributes are qualitative (e.g., text), sometimes values are missing and errors are not basically noise. Errors exists in data for several reasons, e.g., mistypings and misspellings. Moreover, two tuples a, b which differ in their values of attribute Y_i, can still refer to the same real-world object: (1) Usage of abbreviations or alternatives, e.g., *Dr. H. Mueller* and *Hans Müller, Dr.*, (2) Optional elements, e.g., *Hans Müller* and *Hans W. Müller*, or (3) Changing values, e.g., *Hertha Schmidt* might marry *Hans* at some day and be called *Hertha Müller* later on. Many software packages are specialized to find duplicates among address data. They are able to split and standardize *name* and *address* data adequately. Obviously, similar problems arise for other domains and a more generic approach is needed. Given a *Same*-relation, which contains many pairs of tuples with attribute value variations, the influence of variations as described above can be analyzed. Furthermore, if some variations, such as abbreviations, are frequent for an attribute, this knowledge can be utilized, e.g., by use of an adequate distance measure. The inaccuracy of attributes Y might be known and measurable with a distance measure $\text{dist} : \text{dom}\, Y \times \text{dom}\, Y \to \mathbb{R}$. If a value $\Delta \geq 0$ is given, we get the estimator

$$\Delta\text{-accuracy}(Y) := \frac{|\{(a,b) | a \equiv b \wedge \text{dist}(Y(a) - Y(b)) \leq \Delta\}|}{|\{(a,b) \mid a \equiv b\}|}. \tag{17}$$

As special case of (17) we can estimate the *absolute* correctness of an attribute set Y by $\text{accuracy}(Y) := |\{(a,b) \mid a \equiv b \wedge Y(a) = Y(b)\}| / |\{(a,b) \mid a \equiv b\}|$.

Similar to the Δ-accuracy defined above, we can extend the goodness-measure (14) similarly (compare figure 1).

Remark III.14. Note, that for attribute sets Y with average selectivity, e.g. $[\text{selectivity}(Y, A') \geq \frac{1}{10}]$, typically holds $\text{accuracy}(Y, A') \leq \text{goodness}(Y, A')$.

Example III.15 *(BOA cont.).* For a selection of attributes supplied by the BOA–database we performed an estimation of some characteristics c.f. Figure 3. For the Δ-measures we employed suitable distance measures, e.g. the absolute value function ABS( ) for *Size* ($\Delta = 1.0m^2$), *Rooms* ($\Delta = 1$) and *Rent* ($\Delta_1 = 1$. For text-attributes we applied user-defined functions, namely the *Minimum-Edit-Distance* for *Phone* ($\Delta = 2$ edits) and ($1-$*Percentage-Of-Same-Words*) for *FullText* ($\Delta = .1\%$ disagreement). For this estimation we built a sample of pairs.

6) Other Characteristics: **Domains and Content:** Usually a data dictionary is part of a database, where information is supplied for each attribute about the data type, range, format, description etc. Moreover, we can derive empirical characteristics, e.g., frequencies and ranges of values of the attributes of A. Other properties like signatures, fingerprints of attributes (e.g., hash values), n-grams or other codes can be calculated for each attribute from its content, too. But it becomes more difficult to formulate conditions for the hardness involving these characteristics.

Functional Dependencies: The accuracy of functionally dependent attributes can be measured alternatively by the relative frequency of the tuples, where the dependency constraint $[\exists g \, \forall a \in A : g(Y(a)) = X_i(a)]$ failed,

$$\text{FD-accuracy}(Y \cup \{X_i\}) := \frac{|\{a \in A \mid g(Y(a)) = X_i(a)\}|}{|A|}$$

or, alternatively,

$$\Delta\text{-FD-accuracy}(Y \cup \{X_i\}) := \\ = \frac{|\{a \in A \mid \text{dist}\big(g(Y(a) - X_i(a)\big) \leq \Delta\}|}{|A|},$$

for some $\Delta \in \mathbb{R}_{>0}$. Unfortunately, both estimates are limited to sets of functional dependent attributes.

B. The Usage of Characteristics as Weights in the Hardness *Measure*

Next we define the weight function Ψ, to be used in Definition III.1. Let $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_n\}$ the set of *characteristics* supplied with A, and let each class $\mathcal{C}_i$ consist of *characteristics* of the same type, e.g. $\text{duplicates}(A)$. Typically, we have only some information about the *monotony* and *extreme values* for a class of characteristics. We make use of weight functions of its simplest kind, that match these properties, t.i. linear functions. According to Remark III.2 we choose three classes of characteristics, such that each component might compensate the change of other components. Thus —following Claim III.6 and Remark III.14— we exclude the characteristics $\text{null}(Y, A)$

TABLE I

MEASURES AND CHARACTERISTICS FOR THE BOA–DATABASE

	Characteristic/Measure				
Attributes	accuracy$^{\mathsf{S}}$	Δ-accuracy$^{\mathsf{S}}$	goodness$^{\mathsf{S}}$	Δ- goodness$^{\mathsf{S}}$	null
DataSource	.8453		.6288		0
FullText	.4120	.5336	.9887	.9869	0
District	1		.5		0
Street	.8721		.9647		.3324
Size	.9984	1	.7193	.5	0
Rooms	.9965	1	.5338	.5	0
Rent	.9665	.9677	.9605	.9538	0
Phone	.9575	.9807	.9564	.9504	.0389
Floor	.4908		.8838		.5651
Rent, Size	.9654		.9661		0
Rent, Rooms	.9634		.9657		0
Street, Floor	.3971		.9844		.6670
Street, Phone,Floor	.3735		.9968		.6879

$$\Delta\text{-goodness}(Y) \;=\; \frac{\hat{P}\big(\ \text{dist}\big(Y(a),Y(b)\big) \le \Delta \mid a \equiv b\big)\,\hat{P}\big(a \equiv b\big)}{\sum_{\diamond\in\{\equiv,\not\equiv\}} \hat{P}\big(\ \text{dist}\big(Y(a),Y(b)\big) \le \Delta \mid a \diamond b\big)\,\hat{P}\big(a \diamond b\big)} \tag{16}$$

Fig. 1. The Δ-goodness —an extension of the goodness-measure

$$\Psi_1\Big([\text{reduction\_rate}(A) = c]\Big) \;:=\; 1 - c, \tag{18a}$$

$$\Psi_2\Big([c_1 \le \text{duplicates}(A) \le c_2]\Big) \;:=\; \begin{cases} 0 & c_2 = 0 \;\vee\; c_1 = |A| - 1, \\ \theta_1 + (1-\theta_1)\frac{c_2 - c_1}{|A|-1} & \text{otherwise.} \end{cases} \tag{18b}$$

$$\Psi_3\Big(\max_{Y'\subset Y}\big(c:\ [\Delta\text{-goodness}(Y', A) \ge c]\big)\Big) \;:=\; \begin{cases} 2 - 2c & c > \frac{1}{2} \\ 1 & \text{otherwise.} \end{cases} \tag{18c}$$

Fig. 2. The components of the weight function $\Psi(\mathcal{C})$

and accuracy(Y, A) from the weight function Ψ of Definition (3).

Definition III.16 (Weight Function). Let $Y \subset X$ an attribute set. Then the *weight function* of the *hardness*-measure (3) is defined as

$$\Psi(\mathcal{C}) = \prod_{i=1}^{3} \Psi_i(\mathcal{C}_i), \tag{19}$$

whereby Ψ_i denotes the *weight function* for the suggested class of characteristics $\mathcal{C}_i \subset \mathcal{C}$. The weight functions are defined as displayed in figure 2 (θ_1 is chosen from the interval $(0, 1)$, e.g. $\theta_1 = 0.01$). The characteristic goodness is included as special case of the Δ-goodness.

If $\mathcal{C}_i = \emptyset$, we set $\Psi_i(\emptyset) = 1$ (since no reduction of complexity can be achieved with this class of characteristics).

Example III.17 *(BOA cont.).* Let Y be a selection of 13 attributes from the BOA-database. Then we can calculate the hardness measure: (1) hardness$(Y, A, \mathcal{C}) = .8578$ for $\mathcal{C} = \emptyset$, (2) hardness$(Y, A, \mathcal{C}^{\mathsf{S}}) = .1031$ for the constraints calculated precisely at the *Same*-relation, and (3) for several sets of constraints fixed by experts, e.g. hardness$(Y, A, \mathcal{C}) = .8256$ for $\mathcal{C} = \{[\text{reduction\_rate}(A) = .95]\}$ (low-level expert) or hardness$(Y, A, \mathcal{C}) = .5976$ for $\mathcal{C} = \{[\text{reduction\_rate}(A) = .998761],\ [\text{goodness}(Y', A) \ge .95],\ [538 \le \text{duplicates}(A) \le 2,535]\}$ for an attribute set $Y' \subset Y$ (high-level expert).

IV. A Framework for Comparative Analysis

A software benchmark is a prescription for a set of measurements to evaluate some category of software capability, usually performance.

P. O'Neill (in [14, p. 602])

In this section, we describe a framework allowing the evaluation of object identification solutions—specific software packages and algorithms, capable of solving the object identification problem described above. An object identification solution is compounded with its implementation and a fixed hardware/software equipment.

Our objective is similar to the objective of benchmarking of management systems. J. Gray states four important criteria for a good domain-specific database benchmark [15], namely *Relevancy*, *Portability*, *Scalability*, and *Simplicity*. We use these criteria as guiding principles for our framework: (1) Relevancy: Duplicates of the Benchmark database should contain variations that are typical of real world duplicates. Matching of typical data characteristics of real-world databases, e.g., existence of semantic keys, accuracy, (2) Portability: Avoidance of system specific functionality in the general test prescription like *work flows* or *stored procedures* of database management systems, (3) Scalability: Applicability to small and large database sizes $|A|$, (4) Simplicity: Usage of simple-as-possible test data like a database containing a single table using standard data types.

Standard database benchmarks, such as the TPC-benchmark of the *Transaction Processing Performance Council* [16], [15] use artificial data. Because each domain provides its own specifics, it is difficult to generate representative artificial data for object identification problems. For example, errors in name and address data are specific. Therefore we vote for the following *test framework*.

Definition IV.1 (Test Framework). A *Test Framework* for object identification solutions is given as triple $(\mathcal{D}, \mathcal{Q}, \mathcal{S})$, where

- $\mathcal{D}$ is a test database made of a database A and its characteristics $\mathcal{C}$ and a *Same*-relation as introduced in Section III,
- $\mathcal{Q}$ is a definition of quality criteria and description of their computation for a object identification solution on $\mathcal{D}$, c.f. section IV-B, and
- $\mathcal{S}$ is a specification of the test procedure (also called control logic). It consists of a detailed plan prescribing the way to the achieve reproducible results $\hat{\mathcal{Q}}$, allowing exact comparisons of different test runs, c.f. Section IV-C.

A. The Test Database

There exist two possibilities to build a test database:

Real-World Test Database: Given a database with duplicates an domain expert seeks for duplicates and fills in a *Same*-relation, using the *ID*'s of duplicates. For example this was applied to census data, c.f. [17], [18].

Artificial Test Database:[7] A program inserts duplicated records into a given database. The inserted duplicates are randomly produced corrupted and the corresponding *ID*'s are inserted as new tuples into the *Same*-relation. For example this was applied to medical records, c.f. [19].

Both techniques have advantages and disadvantages. Many efforts are required for the creation of the *Same*-relation for a *real-world test database* and the overall number of duplicates is usually limited. For the *artificial test database* —in contrast, merely the error generating process has to be specified to generate an arbitrary large *Same*-relation. The most important advantage of the *real-world test database* is that the variations of data among duplicates are variations as they realistically occur in the real world. Artificially introduced errors, on the other hand, are not domain specific, do not necessarily cover all error types, and their distribution might be unrealistic. The BOA-database contains at least 20% duplicates, thus yielding a real-world test database.

B. Quality of Object Identification Solutions

We suggest the following criteria for the evaluation of the quality of an solution:

- **Quantitative Criteria:** Correctness, Scalability, Performance, and Expenditures/Costs
- **Qualitative Criteria:** Usability, Integrability, Reliability/Completeness, Robustness, Transparency, Adaptability, and Flexibility

We discuss shortly these criteria: (1) *Correctness*: Estimation of misclassification rates for test runs, (2) *Scalability*, e.g. to $|A|$ or w.r.t. the number of undetermined cases and presets misclassification limits, (3) *Performance*: Computational efforts, e.g. complexity of algorithms, computational time for test runs, (4) *Expenditures*: Manual efforts for starting operations, e.g. installation, preprocessing, and efforts for learning (5) *Costs*: Expenses for running operations, e.g. hardware, software licenses, and maintenance (6) *Usability*, e.g. the need of specialized experts and the possibility of automated or incremental updates, (7) *Integrability* into existing software architectures, e.g. interfaces, data/object exchange, remote control, (8) *Reliability/Completeness*, e.g. well-tried, faultless solutions, (9) *Transparency*, e.g. understandability, non-proprietarity of algorithms, heuristics, and results, (10) *Adaptability/Flexibility*: Possibility of automated and incremental updates, e.g. carrying in improved expert/user experience, adaptability to the life cycle of data and the evolution of data models.

If there is a weakness of defining computable measures for some criteria, it might be sufficient to make a qualitative evaluation with a rank or score for these criteria. Note, that this situation can occur for most of the above criteria, but without any test database, the evaluation can never be carried out in a quantitative manner.

Among all the above quality criteria we focus in this article on the *Correctness* criterion. The *correctness* of an object identification solution can be measured by (compare figure 3)

- The *False Negative Rate* or α*–error*: The probability to miss duplicates,
- The *False Positive Rate* or β*–error*: The probability to falsely match non-duplicates.

$$\alpha := P\big(\delta_{\mathcal{C}}(a,b) = 1 \mid a \equiv b\big), \tag{20a}$$
$$\beta := P\big(\delta_{\mathcal{C}}(a,b) = 0 \mid a \not\equiv b\big). \tag{20b}$$

Given the *Same* relation on $\mathcal{D}$, i.e., duplicates are known, these probabilities can be estimated on $\mathcal{D} = (A, \mathcal{C}, \mathit{Same})$ as follows:

$$\begin{aligned}\hat{\alpha} &:= \hat{P}\big(\delta_{\mathcal{C}}(a,b) = 1 \mid a \equiv_{\mathsf{S}} b\big) \qquad (21a)\\ &= |\{(a,b) \mid \delta_{\mathcal{C}}(a,b) = 1 \wedge a \equiv_{\mathsf{S}} b\}| / |\{(a,b) \mid a \equiv_{\mathsf{S}} b\}|,\end{aligned}$$

$$\begin{aligned}\hat{\beta} &:= \hat{P}\big(\delta_{\mathcal{C}}(a,b) = 0 \mid a \not\equiv_{\mathsf{S}} b\big) \qquad (21b)\\ &= |\{(a,b) \mid \delta_{\mathcal{C}}(a,b) = 0 \wedge a \not\equiv_{\mathsf{S}} b\}| / |\{(a,b) \mid a \not\equiv_{\mathsf{S}} b\}|,\end{aligned}$$

If the decision $\delta_{\mathcal{C}}(a,b) = 0/1$ was left open for a portion of ε pairs as discussed in Remark II.1, these pairs should not to be taken into account for the calculation of the error rates.[8]

Remark IV.2. Alternatively to the error rates, we can estimate the measures

$$\begin{aligned}precision &:= \hat{P}\big(a \equiv_{\mathsf{S}} b \mid \delta_{\mathcal{C}}(a,b) = 0\big)\\ &= \frac{|\{(a,b) \mid \delta_{\mathcal{C}}(a,b) = 0 \wedge a \equiv_{\mathsf{S}} b\}|}{|\{(a,b) \mid \delta_{\mathcal{C}}(a,b) = 0\}|}\end{aligned}$$

and

$$recall := \hat{P}\big(\delta_{\mathcal{C}}(a,b) = 0 \mid a \equiv_{\mathsf{S}} b\big) = (1 - \hat{\alpha}).$$

Note, that the undetermined matches decrease only the value of *recall*, while the undetermined nonmatches influence neither the values of *recall* or *precision*. Both measures can be aggregated to the *Match-Accuracy* introduced by Melnik et al. [20], $match\text{-}accuracy := recall\Big(2 - \frac{1}{precision}\Big)$. The *match-accuracy* measures the user effort needed to transform the result of an object identification solution into the *correct* answer, as reported in the *Same*-relation.

[7] As briefly discussed above, we vote *against* the usage of completely artificial data for the test database.

[8] One may argue, that the error rates are only correct, if the value of ε is added to the error rates $\hat{\alpha}, \hat{\beta}$, since these pairs where either accepted or rejected as duplicates. But this is too restrictive.

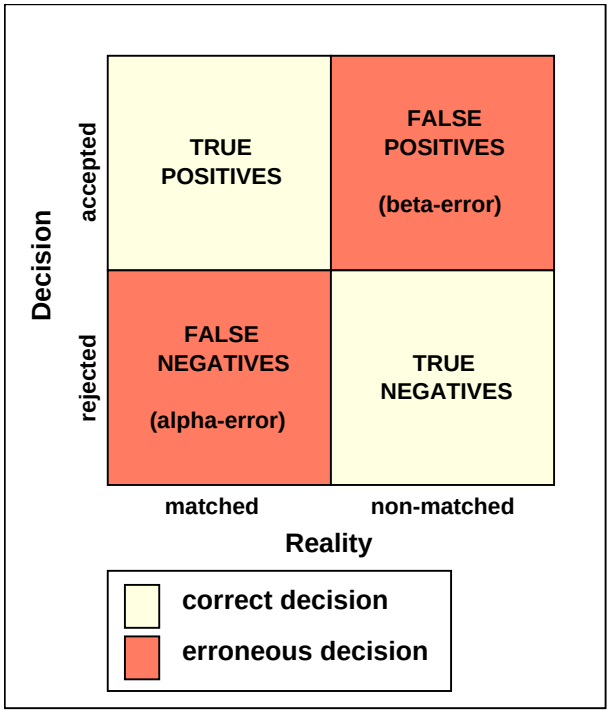


Fig. 3. Decision vs. Reality for object identification solutions

C. The Test Specification

Testing an object identification solution $\delta_{\mathcal{C}}$ accessing $\mathcal{D}$ is a set of test runs according to the specification $\mathcal{S}$, leading to results $\hat{\mathcal{Q}}$ for each test run. The test specification is a guideline for the test of an object identification solution on a given test database. For evaluation, each quality criterion requires a detailed description of the methods for achieving comparable results. To clarify this idea, we document the test specification for the correctness quality criteria en detail. For this specification we adopt the learn-and-test-paradigm of supervised learning:

Given a set of labelled examples, a classification is learned for a subset and tested at the complement set.

In this manner unbiased estimators can be obtained for the correctness. The complete specification is as follows:

1) **Preconditions:** (1) A complete *Same*-relation is supplied with the test database $\mathcal{D}$, (2) A *PreSelection* of pairs from $A \times A$ is provided in order to downsize the size of pairs, c.f. Section III-A.2. (3) The transitive closure $TC(Same)$ of *Same* is given.[9] (4) An empty table *P(PairID, a_ID, b_ID, Same, a_X1,..., a_Xn, b_X1,..., b_Xn)* exists in the database, whereby $\mathrm{dom}(a\_Xi) = \mathrm{dom}(b\_Xi) := \mathrm{dom}(A.Xi)$ and $\mathrm{dom}(Same) = [0,1]$.
2) **Construction of the Samples:** (1) Insert into *P* a random sample of matched pairs[10] of size $N^{(0)} \leq |TC(Same)|$ and set $P.Same = 0$. (2) Insert into *P* a random sample of nonmatched pairs[11] of size $N^{(1)} \leq (|PreSelection| - |TC(Same)|)$ with $P.Same=1$. (3) Update for each $pair \in P$ the attributes *pair.a_Xi*, $i = 1, \ldots, n$ with the values of the respective record in *A*, $a \in A : pair.a\_ID = a.ID$, and for the attributes *pair.b_Xi* respectively, (4) Split up the table *P* into a learning sample *L* (of size N_L) and a testing sample *T* (of size $N_T = N^{(0)} + N^{(1)} - N_L$) and nullify the value of *T.Same*.
3) **Establishment of the Decision Rule:** Then the object identification solution $\delta_{\mathcal{C}}$ is to be established by use of the learning sample *L*, hiding all other records from it, especially the testing sample *T*.
4) **Validation of the Decision Rule:** (1) Apply the decision rule $\delta_{\mathcal{C}}$ to all pairs in the testing sample *T*, and thus the Attribute *T.Same* is valuated for all pairs. (2) Calculate the correctness measures for the decision rule at the testing sample *T* (e.g., the error rates or precision and recall).

Remark IV.3. To apply supervised or unsupervised learning, a comparison of the records *a* with *b*, any $pair \in P$ is made of, has to be performed and the results have to be stored, too. Therefore, a set of suitable comparison functions[12] for pairs of records of A has to be specified and implemented, and for each function a column has to be added to the table *P*. Most learning software, e.g. decision trees, can employ learning only based on this comparison values.
For supervised learning, the software has to be feeded with the modified learning sample $L \subset P$, and the learned classification rule $\delta_{\mathcal{C}}$ is to be stored. In case of unsupervised learning, the result is not yet a decision rule, the decision rule has to be established after post-processing of the results, e.g., the labelling of clusters with *Same* or *Not Same*.
In other cases the decision rule $\delta_{\mathcal{C}}$ might be provided by a domain expert, such that learning is not necessary (e.g. the *Sorted-Neighborhood-Method*, where rules of an equational theory have to be coded separately, c.f. [19], [4]). The above described sampling procedure can be simplified to record sampling (instead of the separate creation of pairs of records), if the execution of the decision rule can be performed directly at the database.

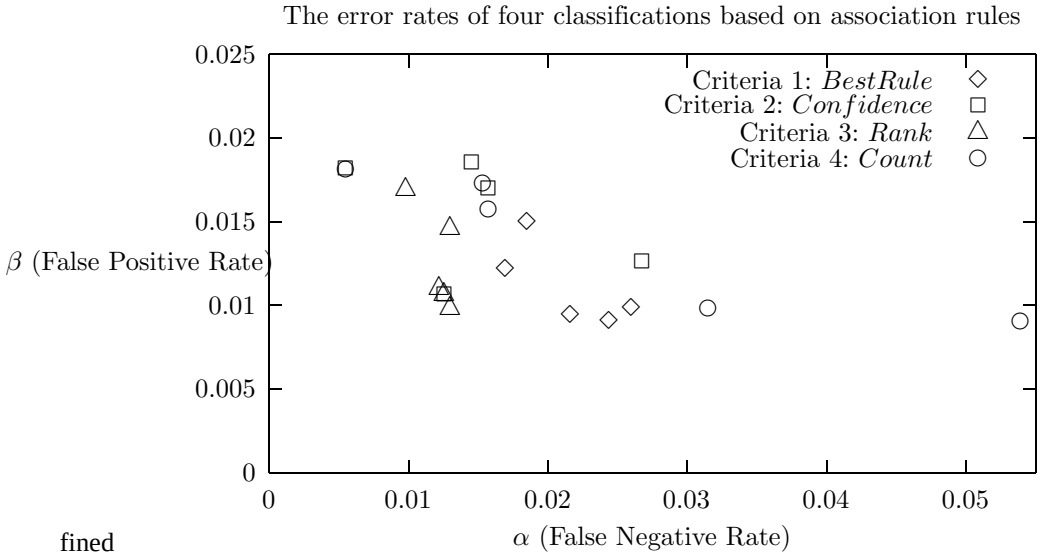
Example IV.4 *(BOA cont.)*. We have applied five successive test runs of association rule mining to equal sized samples of pairs extracted from BOA-database. We have generated approximately 3,000 two- and three-level association rules with minimal support of 1% and minimal confidence of 75%. For at least 300 rules the *Same*-attribute was the decision attribute, in the following named as *positive rules*, if $Same = 0$, and *negative rules*, if $Same = 1$. To classify according to these rules, they must be matched with the comparison values of a pair in the testing sample. Firstly we removed from the matching rules these, which indicate for inter-dependencies

[9] The transitive closure is defined by
$[(\forall a, b \in A : ((\exists a_0 \in A : a < b \wedge (a_0, a) \in Same \wedge (a_0, b) \in Same) \Longrightarrow (a, b) \in TC(Same))]$

[10] A pair *(a,b)* is matched, if $(a,b) \in TC(Same)$

[11] A pair is nonmatched, if $(a,b) \notin TC(Same)$.

[12] e.g. $\mathrm{ABS}(x, y) = |x - y|$, applicable to numeric attributes or the *Minimum-Edit-Distance*(x, y), applicable to text attributes



fined

Fig. 4. The error rates of four object identification solutions calculated at five test samples, all based on association rules extracted from learning samples of pairs selected from the BOA-database (c.f. Example IV.4)

of all but the *Same*-attribute. Nevertheless, often conflicting rules match, such that we needed to apply rule aggregation. Therefore we employed four different rule aggregation criteria, namely (1) *BestRule:* Apply the rule with the largest confidence, (2) *Confidence:* Compute the the sum of the confidence for positive and negative rules, respectively and decide for $\mathit{Same} = 0$ if the first sum is larger than the second, (3) *Rank:* Compute the sums of the inverse rank for positive and negative rules, respectively and decide for $\mathit{Same} = 0$ if the first sum is larger than the second, (4) *Count:* Decide for $\mathit{Same} = 0$, if the number of positive rules exceeds the number of negative rules and for $\mathit{Same} = 1$ else. To avoid undetermined cases, all four criteria provide a clear decision $\mathit{Same} = 0/1$.

The error rates $\hat{\alpha}, \hat{\beta}$ were around 2% for *all* criteria and *all* five test runs, see Figure 4. We deduce from the results, that the variances of the error rates differ. Low variance indicates for the *robustness* of the estimated error-rates, t.i. for new data similar error-rates can be expected. For example, the α-errors of the $Count$-criterion (the circs in figure 4) range from .5% to 5.5%, while the α-errors for the $Rank$-criterion (the triangles) are concentrated at the 1%-line, and the β-errors range from 1% to 1.7%.

The implementation used for this test run was based on Microsoft Visual Basic for Applications, e.g. sampling, comparison and rule aggregation. We performed a lot of dynamic created SQL queries and a huge amount of SQL executions interacting with tables stored in a Microsoft Access database (a few examples are displayed in figure 5). Nevertheless, due to the simplicity and the portability, the above test specification is portable to arbitrary database management systems. A detailed prescription of the technical details for this test specification is currently in preparation.

V. Summary

Quality is an important issue for object identification in databases. We discussed the two main aspects of object identification quality,

- The hardness of object identification problems, and
- The quality of object identification solutions — e.g.correctness— in order to enable a comparative analysis of different solutions at test databases.

We described detailed the determination of the *hardness*, which can be expressed by a set of complexity measures and semantic constraints for an object identification problem. We defined an indicator for the overall hardness based upon these properties.

Further we established a test framework. Several quality criteria are recommended. A prescription of the test specification for the correctness criterion is given. We illustrated our approach with a running example, the BOA-database.

In conclusion, following our approach, it is possible to establish test databases and to evaluate different methods and software packages. We vote for freely available test databases, such that practioners and researchers can perform tests independently from software vendors. Similar to the benchmarks of database management systems, e.g. the TPC-Benchmark [16], comparable results of object identification solutions can be achieved.

References

[1] I. P. Fellegi and A. B. Sunter, "A theory of record linkage," *Journal of the American Statistical Association*, vol. 64, pp. 1183–1210, 1969.
[2] W. E. Winkler, "Matching and record linkage," in *Business Survey Methods*, B. G. Cox, Ed. New York: J. Wiley, 1995, pp. 355–384.
[3] W. E. Winkler, "Record linkage software and methods for merging administrative lists," U.S. Bureau of the Census, Washington D.C., Statistical Research Report Series, 2001.

```
SELECT P.Same AS Same, COUNT(P.Same) AS Count
    FROM P INNER JOIN A ON (P.A_ID = A.ID AND P.B_ID = B.ID)
    WHERE (((1-PercentageSameWords(A.FullText,B.FullText)) <= .05))
    GROUP BY P.Same ORDER BY P.Same;

UPDATE P INNER JOIN A ON (P.A_ID = A.ID AND P.B_ID = B.ID)
    SET P.Phone = DiscreteMinEditDistance(A.Phone,B.Phone,3,3);

UPDATE T SET Same = 0
    WHERE DataSource = 0 AND District = 0 AND ExtraCommission = 2 AND Floor = 2
        AND FreeDate = 0 AND FullText = 0 AND NK = 0 AND Phone = 0 AND Rent = 0
        AND Rent_Brutto = 0 AND Rooms = 0 AND Size = 0 AND Street = 0;
```

Fig. 5. A selection of dynamic created of SQL-statements of the test implementation (`PercentageSameWords` and `DiscreteMinEditDistance` are user defined comparison functions).

[4] M. A. Hernandez and S. J. Stolfo, "Real-world data is dirty: Data cleansing and the merge/purge problem," *Data Mining and Knowledge Discovery*, vol. 2, no. 1, pp. 9–37, 1998.

[5] A. Feekin and Z. Chen, "Duplicate detection using k-way sorting method," in *Proc. of SAC'00, March 19-21, Como, Italy*, 2000, pp. 323–327.

[6] H. Galhardas, D. Florescu, D. Shasha, E. Simon, and C.-A. Saita, "Improving data cleaning quality using a data lineage facility," in *Proc. of DMDW'01, Interlaken, Switzerland*, 2001.

[7] M. Neiling and H.-J. Lenz, "Data integration by means of object identi£cation in information systems." in *Proceedings of the 8th European Conference on Information Systems (ECIS 2000), Vienna, Austria, July 2000*, 2000.

[8] M. Neiling and H.-J. Lenz, "Supplement of information: Data integration by classi£cation of pairs of records," in *Proceedings of the 24th Annual Conference of the Gesellschaft für Klassi£kation, Passau, Germany, March 15– 17, 2000*. Spinger-Verlag, 2000.

[9] "The INTEGRITY^{TM} software for record linkage," formerly AUTOMATCH-software; available from http://www.ascentialsoftware.com.

[10] "The MERGE/PURGE component of the CENTRUS^{TM} data quality software," available from http://www.centrus.com/.

[11] "The FUZZY! POST^{TM} software for address standardization," available from www.fuzzy-informatik.com.

[12] E.-P. Lim, J. Srivastava, S. Prabhakar, and J. Richardson, "Entity identification in database integration," *Information Sciences*, vol. 89, no. 1, pp. 1–38, 1996.

[13] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, April 1997.

[14] P. O'Neill, *Database Management*. Morgan-Kaufmann, 1994.

[15] J. Gray, Ed., *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*. Morgan Kaufmann, 1993.

[16] "The TPC benchmarks," c.f. http://www.tpc.org.

[17] M. A. Jaro, "Advances in record-linkage methodology as applied to matching the census of Tampa, Florida," *Journal of the American Statistical Association*, vol. 84, no. 406, pp. 414–420, 1989.

[18] W. E. Winkler, "Quality of very large datbases," U.S. Bureau of the Census, Washington D.C., Statistical Research Report Series RR2001/04, 2001.

[19] M. A. Hernandez, "A generalization of band joins and the merge/purge problem," Ph.D. thesis, Columbia University, 1996.

[20] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity ¤ooding: A versatile graph matching algorithm and its application to schema matching," in *Proc. 18th ICDE Conference*, 2002.