Journal of King Saud University – Computer and Information Sciences 32 (2020) 126–136

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Methodology for fuzzy duplicate record identification based on the semantic-syntactic information of similarity

Djulaga Hadzic *, Nermin Sarajlic

Faculty of Electrical Engineering, University of Tuzla, Franjevacka 2, 75000 Tuzla, Bosnia and Herzegovina

ARTICLE INFO

Article history:
Received 22 November 2017
Revised 19 April 2018
Accepted 3 May 2018
Available online 12 May 2018

Keywords:
Duplicate records
Fuzzy
Semantic
Syntactic
Blocking
Windowing

ABSTRACT

There are different methodologies for identification of fuzzy duplicate records in the process of data cleaning for data warehouse and data mining. The methodologies for duplicate record identification can be classified into three groups: blocking methods, windowing methods, and semantic methods. The article specifically focuses on semantic methods and describes Semantic-Syntactic Method for fuzzy duplicate record identification. Based on the conducted testing, comparative analysis is presented of the results obtained through the Semantic-Syntactic Method and two other standard methods over a selected data set. In the end, the article presents conclusions with regard to the quality and efficiency of the Semantic-Syntactic Method, as well as suggestions for future research in this field.

1. Introduction

With each increase in the amount of data and with each improvement of information systems in data collection from distributed and diverse sources, the amount of problems related to the quality of data also increases. Information and data quality are the topics of extensive and very active research both from the perspective of information systems and databases. Databases play an important role in today's IT-based economy. Many industries and systems depend on the accuracy of databases to carry out operations (Elmagarmid et al., 2007). As database owner, one must also be confident that there are no duplicate records in a database where different identifiers are used for the same entity. This situation is however common, for example in customer databases, where the same customer can have several records due to name variations or address changes (Christen, 2012). The problem of finding records that relate to the same entities not only applies to databases that contain information about people. Other types of entities that sometimes need to be matched include records about businesses, consumer products, publications and bibliographic citations, Web pages, Web search results, or genome sequences (Christen, 2012).

The data quality was defined by Tayi and Ballou (1998) as "usefulness of data" i.e. as appropriateness of their use. One of the most interesting problems related to data quality is multiple, and at the same time, in real world, different representation of the same objects in data. These multiple data records, also known as duplicates, are very difficult to identify, especially in big data. Simultaneously, they reduce the usefulness of data, cause unnecessary costs, customer dissatisfaction, inaccurate performance indicators, and prevent the appropriate understanding of data and their values. Duplicate records in the context of this study do not represent their identical copies (replicas) but rather they represent slight, and sometimes even significant, differences in individual data values. Therefore, these duplicate records are called fuzzy duplicate data records (Maletic and Marcus, 2000). In the context of the traditional Data Base Management System (DBMS), duplicate data records represent identical copies (Hernandez and Stolfo, 1995). These duplicate records are easy to identify through simple sorting of data in all the columns, and they are not the topic of this study. The topic of this study are fuzzy duplicate records.

Duplicate record detection and elimination is a challenge of data cleaning (Ripon et al., 2010). Duplicate detection is a very expensive operation, as it requires the comparison of every possible pair of duplicates using the typically complex similarity

* Corresponding author.
E-mail addresses: dulaga.hadzic@student.fe.untz.ba (D. Hadzic), nermin.sarajlic@untz.ba (N. Sarajlic).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2018.05.001
1319-1578/

measure (Weis et al., 2008). The identification of duplicate records must solve the mutually linked (inherent) difficulties (Naumann and Herschel, 2010): find all duplicate records in all data sets quickly (efficiency) and correctly identify duplicate records and those that are not (effectiveness). The first difficulty in identification of duplicate records is a square complexity of a problem. Conceptually, each record has to be compared to every other record. The general idea is to avoid the comparison of considerably different records and to concentrate only on comparison of records having, within the default limits, quickly identified similarities. Another difficulty of duplicate record identification is the defining of an appropriate level of similarity based on which it will be determined whether the two records are matched or not. The similarity threshold has to be chosen so it classifies the pairs of records as duplicate or different.

Most previous research in this area is mainly concerned with the adaptation of existing methods of identifying duplicate records based on sorting a particular key and placing them in blocks (partitions) or by using a sliding window over sorted records. In both groups of methods, the participation of a man in the execution of the method is not excluded and in methods based on the equational theory of role of man in the execution of the method is indispensable. In this paper, a new approach to duplicate record detection is proposed. A new approach is based on a contextual, straightforward selection of records pairs to compare and exclude a person in executing the method through the automation of those processes.

2. Methodologies for duplicate record identification

The most popular methods for identification of duplicate records are the methods of partition, namely blocking methods and windowing methods (Draisbach and Naumann, 2011). Recently, there are methods that contain semantic elements. Such methods are called semantic methods.

2.1. Blocking methods

The main idea in blocking is to group similar records together, called blocks or clusters, using available information from the records. Blocking methods perform a grouping of a set of records into separated partitions or blocks, and then they compare all pairs of records only within individual blocks (Maletic and Marcus, 2000; Maydanchik, 1999). Because real-world data are often dirty and contain variations and errors, an important criteria for a good blocking key is that it can group similar values into the same block (Christen, 2012). A good blocking key is critical to the success of a blocking method and will ideally result in lot of small blocks (Gu and Baxter, 2004).

Blocking is achieved by sorting the records according to one or according to a combination of several attributes such as the blocking key, based on which, the records are then separated into mutually exclusive partitions. Therefore, each record will be assigned to only one block. It is desirable to select the blocking keys that will lead to the creation of blocks of approximately the same size. To identify duplicates that differ in the blocking key, a multi-pass method is employed, where for each pass a new blocking key is defined. At the end of this method, a transitive closure is performed over all discovered duplicate records.

The best-known blocking method is the Standard Blocking Method (SBM). The Standard Blocking Method is based on probability techniques. Its implementation commenced with the ideas of duplicate record identification of Newcombe et al. (1959), which were formalized in the mathematical model by Fellegi and Sunter (1969). This method is based on the EM (Expectation Maximization) algorithm (Hylton, 1996; Maydanchik, 1999), proposed by Jaro (1989) to estimate the probability that a record pair belongs to class M; $P(x_i = 1|M)$. The probability that a record pair does not match, i.e. there is a high probability that it belongs to class U, $(x_i = 1|U)$, can be estimated by inspecting a certain number of randomly selected record pairs.

2.2. Windowing methods

Windowing methods sort the data according to the sorting key, and, by using the sliding window, they go through the sorted data and compare only the records within a window. A disadvantage of this approach is that the window size is fixed and difficult to configure: If it is selected too small, some duplicates might be missed (Draisbach et al., 2012). On the other hand, a too large window results in many unnecessary comparisons (Draisbach et al., 2012). The size of the sliding window is expressed in the number of records. In windowing methods, the important factor is the selection of the appropriate size of the sliding window. It is desirable to select the window size in order to compare similar records, i.e. not to compare records that are mutually significantly different. To increase the possibility of the comparison of the records that differ in the sorting key, a multi-pass method is also employed in this case, where for each pass a new sorting key is defined.

The best-known windowing method is the Sorted Neighborhood Method (SNM) by Hernandez and Stolfo (Hernández and Stolfo, 1998; Weis et al., 2008). This method uses rule-based record comparison approach. Record comparison rules are created according to attributes (or combination of attributes), and based on them, record classification is performed. In this method, the used variant of rule-based record comparison is based on the equational theory (Hernández and Stolfo, 1998; Yan et al., 2007).

2.3. Semantic methods

Among the mentioned groups of duplicate record identification methods, there have recently been an increasing number of methods that contain semantic elements, i.e. approaches based on meaning or context of the content or certain relationships between the analyzed data.

Semantics is the theory of meaning http://lingvo.info/en/babylon/semantics. It is a colorful discipline, encompassing various levels and concepts of meaning, be it lexical, formal, structural and functional http://lingvo.info/en/babylon/semantics. Formal semantics attempts to reduce factual parts of sentences to well-defined value systems. Structural semantics treats language as a system relation network.

Classification is the core and basis of semantics. Through the implementation of these duplicate record identification methods, certain modules of semantic classification and relation as well as reduction of content to particular defined value systems have been built in. These techniques commonly combine both semantic and syntactic similarities into one overall similarity value which is used to detect similar entities (Christen, 2012).

3. The Semantic-Syntactic method

Through their scientific work, the authors of this paper have developed one of the duplicate record identification methods. The working title of this method is the Semantic-Syntactic Method (SSM). This method is based on the semantic information of similarity and context. Apart from the elements based on context, the method also contains the syntactic elements from blocking and windowing methods. In the Semantic-Syntactic Method, as a result of its own research, the syntactic elements have been improved or even

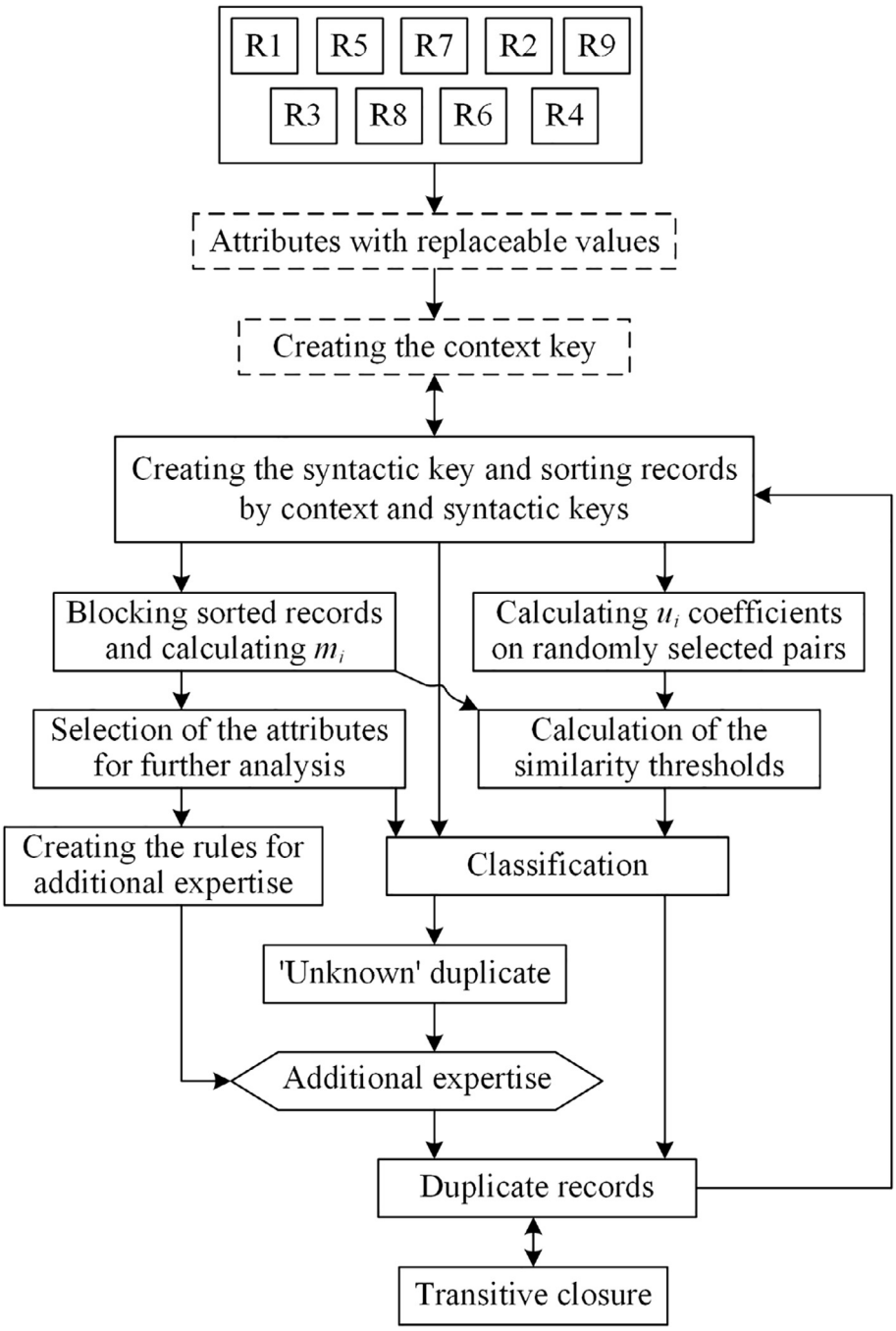

Fig. 1. The basic structure of the Semantic-Syntactic Method.

extended by the context part in order to get the better results. Fig. 1 presents the basic structure of the Semantic-Syntactic Method.

The Semantic-Syntactic Method, with examples of testing over the data set *Dataset1*[1] from the Febrl database, is described in continuation. The basic steps in the implementation of the Semantic-Syntactic Method are:

3.1. Attributes with replaceable values

During data update, it is possible to input a value of one attribute as a value of another attribute and vice versa. For example, the values for the *surname* and *given_name* attributes can be replaced and the values of *address_1* and *address_2* attributes can be replaced as well. In this method, when two records are compared, two types of comparison are conducted, one related to the attributes with the same values and the other related to attributes with changeable values.

3.2. Context key creation

The construction of the record classification semantic system begins with the construction or definition of the context key which will be used to separate different records into different record groups. In our testing to calculate the value of the context key, the attribute of the *postcode* is used. The calculating value of a

[1] http://bih5.net/nir/febrl-dataset1.zip (downloaded from https://github.com/J535D165/FEBRL-fork-v0.4.2/tree/master/dsgen)

Table 1
The example of the calculation value of the context key for the *Dataset1*.

Rec_id	Postcode	Similarity with previous value	The value of the context key
rec-334-dup-0	2000	–	1
rec-334-org	2000	Sim('2000','2000') = 1 > T_k	1
rec-256-dup-0	2006	Sim('2000','2006') = 0.83 > T_k	1
	...	...	...
rec-101-org	3987	Sim('3987','3986') = 0.83 > T_k	1
rec-223-dup-0	4011	Sim('4011','3987') = 0 < T_k	2
rec-223-org	4011	Sim('4011','4011') = 1 > T_k	2
	...	...	...

context key is performed with the calculation of the similarity of neighboring values of the expression (*postcode*) according to which the records are sorted. The result of the comparison of similarity of values of the expression for calculating the context key is compared to the defined threshold of similarities (T_k). The value of the first record's context key is 1, and each next record is given a context key with the value calculated on the basis of the comparison of value of the expression for calculating the context key with the previous record. The change of the context key's calculated value of one record (the increase of previous context key value by 1) compared to its precursor is preformed if the similarity between attribute value, for the context key calculation, is below the value of a defined similarity threshold (T_k). Table 1 gives the example of calculating the value of the context key for the *Dataset1* for the lower threshold of similarity $T_k = 0.04$.

3.3. Syntactic key creation

Syntax key causes similar records to be closer to each other. This method uses a multiple number of passes to analyze records according to different syntax keys. A separate syntax key is defined for each pass. A defined expression for calculating the syntactic key value can contain a value of the whole attributes or their parts. Besides the standard way, in a syntactic key, there is a possibility to apply the operator or functions for the separation of particular attribute values on the basis of a particular pattern. After the syntactic key is created and added to the records, the records are sorted according to the context key and syntactic key. Examples of calculating the syntax key value for the *Dataset1* in first passage, are given in Table 2.

3.4. Calculation of the coefficients of congruency probability m_i and non-congruency probability u_i

Coefficient m_i represents congruency probability and u_i represents non-congruency probability between attributes of two records. Blocks of records within which the records are mutually compared are created over a particular number of records, sorted according to the context key and syntactic key, by comparing similarities of neighboring records' syntactic keys with the defined similarity threshold. For each pair of records compared within the same blocks, a calculation of similarity vectors γ_j (j is index of a pair of records) is performed through a binary model (Jaro) for the calculation of similarity vector value. After performing the comparison of all record pairs within blocks, the formed similarity vectors are grouped and stored with their frequency counter $f(\gamma_j)$ for the calculation of values m_i and u_i of all record pairs. Table 3 gives an example of calculating the similarity vectors and their frequency counters over the *Dataset1* in first passage.

The calculation of m_i and u_i values is done from the value of γ_j similarity vectors and their frequency counters $f(\gamma_j)$ through the application of the EM algorithm (Maydanchik, 1999) which generates the m; u and p parameters through the repetitive calculation of E-step and M-step. Examples of calculated values of m_i per passages are given in Table 4.

Although the u_i parameters are calculated through the EM algorithm, the alternative calculation of u_i values is performed over a randomly selected cluster of records that compare themselves mutually and form similarity vectors. According to these similarity vectors and their frequencies, the calculation of u_i parameters value is performed.

Table 2
The example of the calculation value of the syntactic key for the *Dataset1*.

Rec_id	The expression for calculating the syntactic key	The syntactic key value
rec-99-org	*surname & given_name*	*akroyd imogen*
rec-99-dup-0	*surname & given_name*	*akroyd imzogen*
rec-1-dup-0	*surname & given_name*	*alderson karli*
rec-1-org	*surname & given_name*	*alderson karli*

3.5. Attribute selection

Based on the calculated m_i and the defined lower T_m threshold, the selection of attributes for further analysis of the similarity of records is performed. The attributes used for further analysis are those for which the value is defined using (1):

$$m_i \geqslant T_m \tag{1}$$

In this way those attributes that have a large number without values, are eliminated from further analysis. Such attributes can contribute to the wrong classification of two records. The selection of the attributes for further analysis, for each of the passages, is given in Table 4.

3.6. Defining the rules for additional expertise

Based on the calculated m_i values, the rules that will be used for additional expertise for the record pairs which will be classified as unknown (i.e. it is unknown whether the record pairs are duplicate or not), are automatically defined. The rules are defined using semantic similarity operators: *similar*, *verysimilar* and *same*. Based on the semantic similarity operators and m_i values for the selected attributes, it is determined whether the record pairs will be categorized as matched or distinct.

The Eq. (2) shows the conditions under which a selection of similarity operator for i attributes in the j and $j + 1$ records is performed. The selected similarity operator will be applied to the attribute values $v(a_{i,j})$ and $v(a_{i,j+1})$. All possible combinations for obtaining the statement I_i for the value of i attribute can be written as follows:

$$I_i = \begin{cases} v(a_{i,j})similar\, v(a_{i,j+1}), similar \leqslant m_i < verysimilar \\ v(a_{i,j})\, verysimilar\, v(a_{i,j-1}), verysimilar \leqslant m_i < same \\ v(a_{i,j})same\, v(a_{ij-1}), m_i \geqslant same \end{cases} \quad (2)$$

The rule P_{pr} for the assigned passage pr represents a set of previously defined statements mutually connected by Λ operator (Logical AND). $P_{pr} = I_1 \Lambda I_2 \Lambda \ldots I_n$, where n is the number of statements formed for the current passage. It is important to note that the attributes used to develop a syntactic key are not included since it is very likely that the value of these attributes is very similar. If the application of the rule P_{pr}, the additional comparison of the two possibly matched records, gives *True* as a result, then the two records are categorized as duplicate. The automatically created rules are shown in Table 4 for each of the passages.

3.7. Calculation of the similarity thresholds

The similarity thresholds are used to classify pairs of records. Based on the calculated coefficients of congruency probability m_i and non-congruency u_i probability the lower (T_μ) and upper (T_λ) similarity thresholds are calculated using the EM algorithms. T_λ is calculated according to the expression (3), while T_μ is calculated according to the expression (4).

$$T_\lambda = \frac{m(\gamma_{n'})}{u(\gamma_{n'})} \quad (3)$$

$$T_\mu = \frac{m(\gamma_n)}{u(\gamma_n)} \quad (4)$$

Parameter μ represents the rate of wrongly matched record pairs, and parameter λ represents the rate value of wrongly non-matched records. The lower and upper similarity threshold values for each of the passages are given in Table 4.

3.8. Classification

Based on the upper and lower similarity thresholds, the record pairs can be classified as: matched A_1 (duplicate), possibly matched A_2 (unknown) and non-matched A_3 (distinct).

For the fixed rate values of wrongly matched (μ) and wrongly non-matched (λ) record pairs (a, b), Fellegi and Sunter (Lee and Ling, 1999) define the optimum rule of connecting records into Γ (set of all possible matched vectors) on levels μ and λ, marked with $L(\mu, \lambda, \Gamma)$ as a rule where $P(A_1|U) = \mu$, $P(A_3|M) = \lambda$, and $P(A_2|L) \leq P(A_2|L')$ for all other L' rules. Let the relation $m(\gamma)/u(\gamma)$ be sorted in decreasing order and the corresponding γ indexed with $1, 2 \ldots N_\Gamma$.

If $\lambda = \sum_{i=n'}^{N_\Gamma} m(\gamma_i)$ *and* $\mu = \sum_{i=1}^{n} u(\gamma_i)$, $n < n'$, then the optimum rule represents a function of possibilities $m(\gamma)/u(\gamma)$ ratio and it is presented by the following expressions:

$$(a,b) \in A_1 \text{ if } T_\mu \leqslant \frac{m(\gamma)}{u(\gamma)} \quad (5)$$

$$(a,b) \in A_2 \text{ if } T_\lambda < \frac{m(\gamma)}{u(\gamma)} < T_\mu \quad (6)$$

$$(a,b) \in A_3 \text{ if } \frac{m(\gamma)}{u(\gamma)} \leqslant T_\lambda \quad (7)$$

The calculation of the total weight of the record pairs is obtained by summing up all the weights of their n attributes ($i = 1,\ldots,n$). If the attribute i from the record pair is matched, the weight of this attribute is presented by $w_i = log_2(m_i/u_i)$. On the other hand, if the attribute i is not matched, then its weight is presented by $w_i = log_2((1 - m_i)/(1 - u_i))$. If the weight of the record pair comparison is above the upper threshold (T_μ) than this record pair is truly matched (A_1). If the calculated weight of the compared record pair is between the upper (T_μ) and lower (T_λ) similarity threshold, then this record pair is classified as possibly matched (A_2). For the record pairs classified as possibly matched (A_2), an additional expertise has to be done.

In this method, the selection of the next pair for comparison depends on the result of the previous pair of records Fig. 2). Thus, the comparison begins with the comparison of the syntactic keys of the first (R1) and the second record (R2) within the sorted set of data according to the context and syntactic keys for the assigned passage. If the compared pair of records is classified as A_1, they will be put in Cluster 1. R2 and R3 records are going through the process of comparison as R1 and R2 records and, according to that, this record pair is classified as A_1, A_2, or A_3. In Fig. 2, successful value comparisons are presented with the full line, and the unsuccessful ones with the dashed line. The comparisons between records are, as presented in Fig. 2, performed by step $k = 1$. This means that after the first negative result of the comparison of neighboring records' syntactic key values, a sequence of successful comparisons and a formation of a cluster is interrupted which initiates another sequence of syntactic key value comparison of records and formation of a new cluster of mutually connected records.

Record pairs under a1, a2 and a3 Fig. 2 are duplicate records identified through the comparison of neighboring records in clusters 1, 2 and 3. After creation of one cluster, and before the start of the new comparison between neighboring values, the semantics of transitional closing between the records within that cluster is used (b1, b2 and b3 in Fig. 2.

3.9. Additional expertise

For those record pairs that remain classified as possibly matched (A_2) after all passages, it is necessary to perform the additional analysis in order to determine whether these record pairs are matched (A_1) or distinct (A_3). Due to the above mentioned, this new method implements a part whose aim is to automatically define a set of rules over the attributes of data set (described in step 6) based on which it is determined whether the possibly matched record pairs A_2 are actually matched (the same) or not. Depending on the passage in which the record pair is classified as possibly matched A_2, the rule defined for that passage is applied to that pair. If the rule applied to a certain record pair which is classified as possibly matched gives the result *True*, then the mentioned record pair is classified as A_1, i.e. matched (the same).

3.10. Transitive closure

Transitive closure over the already defined record pairs which are classified as matched, either in the regular comparison or applying the rule over the possibly matched records, is performed only on record pairs identified as such through different passes.

Table 3
The example of the vectors and their frequency counters for the *Dataset1*.

Vectors(γ_j)	Frequency $f(\gamma_j)$	Vectors(γ_j)	Frequency $f(\gamma_j)$	Vectors(γ_j)	Frequency $f(\gamma_j)$
1111111111	124	0100000100	26	1110111111	13
0100000000	78	0100000010	25	0110000000	11
1101111111	30	0111111111	17	1111111101	11
1111011111	28	1111101111	14	1110011111	10

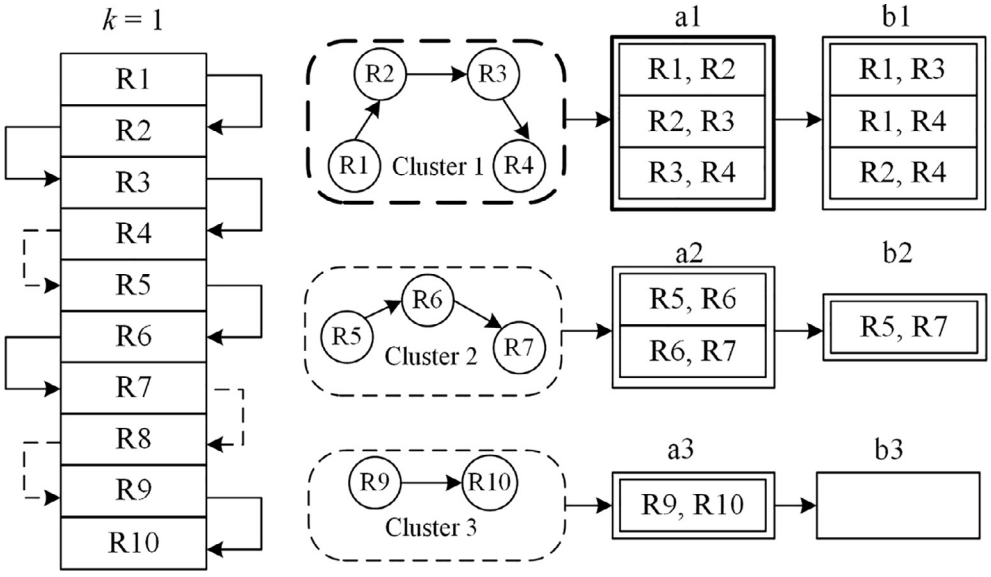

Fig. 2. Selection and comparison of record pairs in the Semantic-Syntactic Method.

This is considered very significant from the aspect of decreasing a number of record pairs which are inspected with the aim of finding new record pairs that are matched.

4. Testing results

The Semantic-Syntactic Method as well as the Standard Blocking Method and the Sorted Neighborhood Method were implemented in the web application *Multiple Records Identification System* (MRIS) (Hadzic et al., 2015). The testing of all three methods, was performed on the same computer platform with Intel™ Core™ i7, 64-bit CPU (2.2 GHz), 16 GB of RAM and Windows 8.1 operating system. For the testing methods used are two sets of data. In addition to the aforementioned data set *Dataset1*, the *Cora1*[2] data set from the Cora database was also used for testing. Cora database offers real data. The data set *Dataset1,* taken from FEBRL database, is synthetic and contains 1000 records of which 500 are duplicate records. The data set *Cora1* contains 500 records of which there are 7821 combinations of record duplicates. The *Dataset1* represents a well-connected data set, and the *Cora1* is a weakly-linked data set.

Analysis of the quality of the results was carried out using standard measures: *Recall*, *Precision* and *F-score* (Branchman and Anand, 1996):

$$Recall = \frac{tp}{tp + fp} \tag{8}$$

$$Precision = \frac{tp}{tp + fn} \tag{9}$$

$$F - score = 2\frac{Recall \cdot Precision}{Recall + Precision} \tag{10}$$

where *tp* stands for *true positives* (correctly identified duplicate records); *fp* are *false positives* (incorrectly identified as duplicate records) and *fn* are *false negatives*. *F-score* is a parameter that gives more weight to either *Recall* or *Precision*. In this article neutral parameterization was used, where *Recall* and *Precision* have same weight. Additional analysis of efficiency and effectiveness of methods is realized through the execution of the individual phases (Time) and the type of the execution phase (Type). Types of execution phases can be *Automatic* and *Manual* (Hadzic et al., 2015). *Automatic* execution means that a method's phase is performed fully automatically by the computer, while *Manual* execution means that a method's phase is entirely performed by a man. The execution time is calculated in total and per phases. The similarity measure (function) *Jaro*, used for the testing, was empirically determined as the best quality one for the selected data set (Hadzic and Sarajlic, 2016).

The following shows detailed testing data for the new Semantic-Syntactic Method, using the MRIS application, over both testing data's sets. A summary and a comparison of the results of all three methods over both testing data's sets using MRIS application are given at the end of this section.

4.1. Testing the Semantic-Syntactic method over the Dataset1

The input parameters and testing phases of the Semantic-Syntactic Method through MRIS application over the dataset *Dataset1*:

1. Defining attributes that can replace its value.
   (*given_name* and *surname*), (*address_1* and *address_2*)
2. The context key and sorting of the data set. (Time: 323.8268 ms Type: Automatic)
   Context key: *postcode*
   The lower threshold of similarity for calculating the context key: 0.40
3. Number of passages and syntactic keys: 3
   Passage 1 – Syntactic key 1: *surname & given_name*;
   Passage 2 – Syntactic key 2: *state & suburb & address1*;
   Passage 3 – Syntactic key 3: *postcode & soc_sec_id & date_of_birth(1,4)*;
4. Defining input values for calculating m_i, u_i, and for the selection of attributes:
   Number of records for blocking: 1000;
   The threshold of similarity for syntactic keys: 0.70;
   The threshold of similarity for m_i: 0.75; The threshold of similarity for u_i: 0.60;
   The lower threshold T_m for the selection of attributes: 0.3
5. Defining input values for creating rules for additional expertise:

[2] http://bih5.net/nir/cora-cora1.zip (downloaded from https://github.com/usc-isi-i2/lsh-linking/tree/master/cora-dataset/cora-utexas)

Lower thresholds for similarity operators: *same* = 0.95; *verysimilar* = 0.80; *similar* = 0.70;

6. Passages
   Table 4 presents the results of testing for the data set *Dataset1* for each of the passages (see in Appendix A).
   The total number of duplicate records after 3 passages is 497.
   The total number of duplicate records after 3 passages for additional expertise is 0.
7. Additional expertise. (Time: 0 ms Type: Automatic)
   Note: Through Passage 2, two pairs of records were identified as A_2. These two pairs of records were identified as A_2 through Passage 3, and then automatically classified as A_1.
   Duplicates through additional expertise: 0
8. Transitive closure. (Time: 832.6932 ms Type: Automatic)
   Duplicates through a transitional closure: 0
9. Standard measures of quality:
   tp = 497 *fp* = 0 *fn* = 3
   Precision = 1 *Recall* = 0.9939879759 *F-score* = 0.99698492462

4.2. Testing the Semantic-Syntactic method over the Cora1

The input parameters and testing phases of the Semantic-Syntactic Method through MRIS application over the data set *Cora1*:

1. Defining attributes that can replace its value.
   No attributes that can replace its value in the data set *Cora1*.
2. The context key and sorting of the data set. (Time: 129.0283 ms Type: Automatic)
   Context key: *Year&Title&Adress&Venue&Class*('\d{4}')
   Note: The pattern (\d{4}) over the combination of attributes for the creation g of context key is a four-digit number that represents the year of publication.
   The lower threshold of similarity for calculating the context key: 0.70
3. Number of passages and syntactic keys: 3
   Passage 1 – Syntactic key 1: *title & author*;
   Passage 2 – Syntactic key 2: *author & title*;
   Passage 3 – Syntactic key 3: *year('\d{4}') & addres('\d{4}') & venue & title(1,5)*;
4. Defining input values for calculating m_i, u_i, and for the selection of attributes:
   Number of records for blocking: 500;
   The threshold of similarity for syntactic keys: 0.70;
   The threshold of similarity for m_i: 0.75; The threshold of similarity for u_i: 0.60;
   The lower threshold T_m for the selection of attributes: 0.3
5. Defining input values for creating rules for additional expertise:
   Lower thresholds for similarity operators: *same* = 0.95; *verysimilar* = 0.80; *similar* = 0.70;
6. Passages
   Table 5 presents the results of testing for the data set *Cora1* for each of the passages (see in Appendix A).
   The total number of duplicate records after 3 passages is 6753.
   The total number of duplicate records for additional expertise is 1.
7. Additional expertise. (Time: 21.5499 ms Type: Automatic)
   Table 6 presents the pairs of records, that have been identified through the previous passages as a "possible match" (A_2) by their identifiers, a similarity vector, a calculator weight and a passage in which they have been identified.
8. Transitive closure. (Time: 837.629 ms Type: Automatic)
   Duplicates through a transitional closure: 198
9. Standard measures of quality:
   tp = 6952 *fp* = 0 *fn* = 869
   Precision = 1 *Recall* = 0.888874680306905 *F-score* = 0.941168505856069

4.3. The comparison of the results testing for three methods

Tables 7 and 8 (see in Appendix A) provide the summary and by stages of execution compared test results for all three tested methods using the application MISR for data sets *Dataset1* and *Cora1*, respectively. The cells in Tables 7 and 8 filled with 'X', mean that the certain method in its implementation does not contain that part of phase or result.

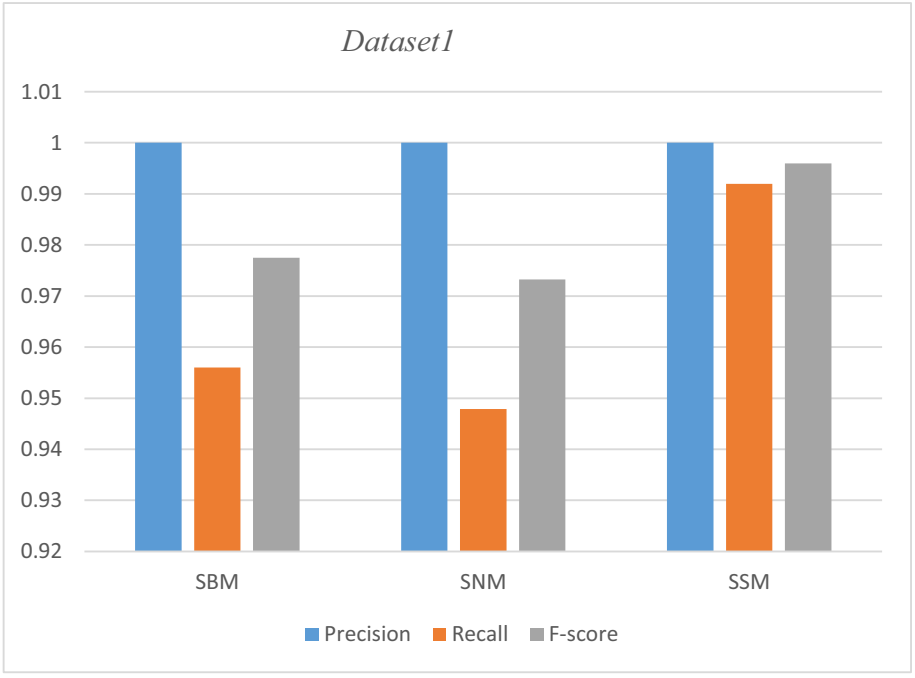

Fig. 3. The comparison of the results for the quality of the tested methods over the Dataset1.

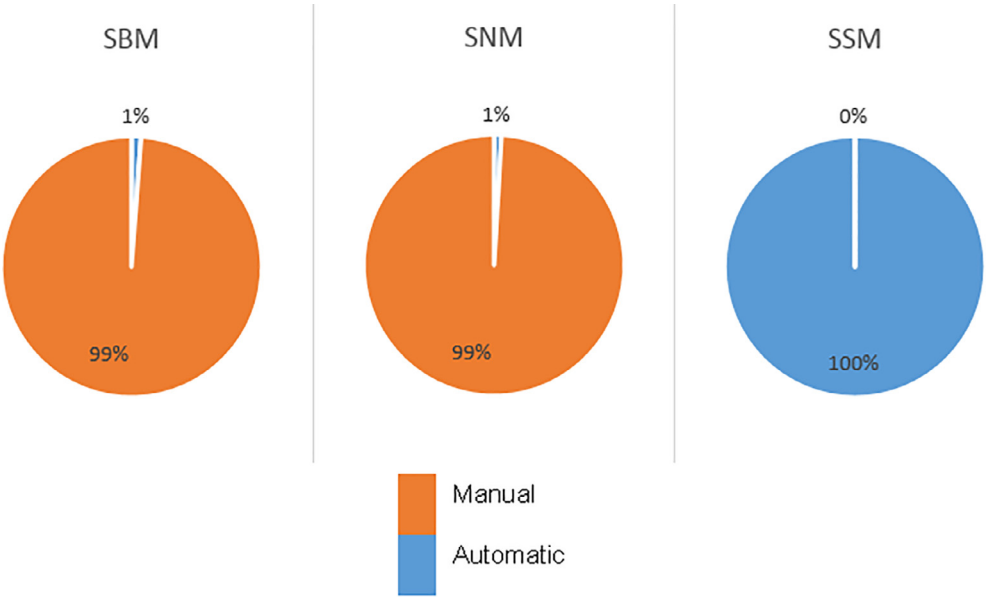

Fig. 4. The comparison of execution time of the tested methods over the *Dataset1*.

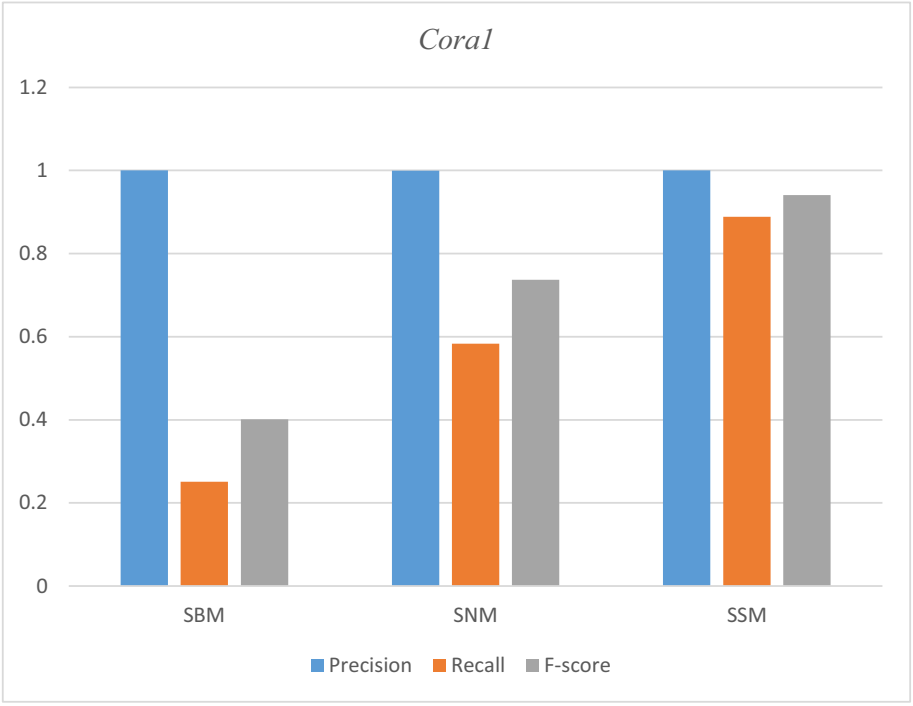

Fig. 5. The comparison of the results for the quality of the tested methods over the *Cora1*.

5. Discussion

The previous part (Section 4) presents the detail testing results of the SSM method over two data sets Dataset1 and Cora1. Table 7 presents the summary and compares the three methods' testing results for the *Dataset1*. The common thing for the SBM and SSM methods is the identification of new pairs of records through passages as possibly matched or unknown. Additional expertise is necessary for such record pairs, according to which it is possible to identify new duplicate records. There is no additional expertise in the SNM method Table 7 – field filled with X). From the aspect of effectiveness, the best testing results over the data set *Dataset1* were obtained by the SSM method. Therefore, the SSM method gives the best results, followed by the SBM method and finally the SNM method, with the lowest results of the *F-score* Fig. 3).

The efficiency of the methods was measured by the total time spent on conducting the testing and the time spent on each phase of execution. Based on the measured time, it is possible to validate the human role (*Manual*) in their execution. According to the test results presented in Table 7, the SBM and the SNM methods are time-consuming in *Manual* execution phases Fig. 4). The reason for this is a relatively large number of record pairs analyzed by

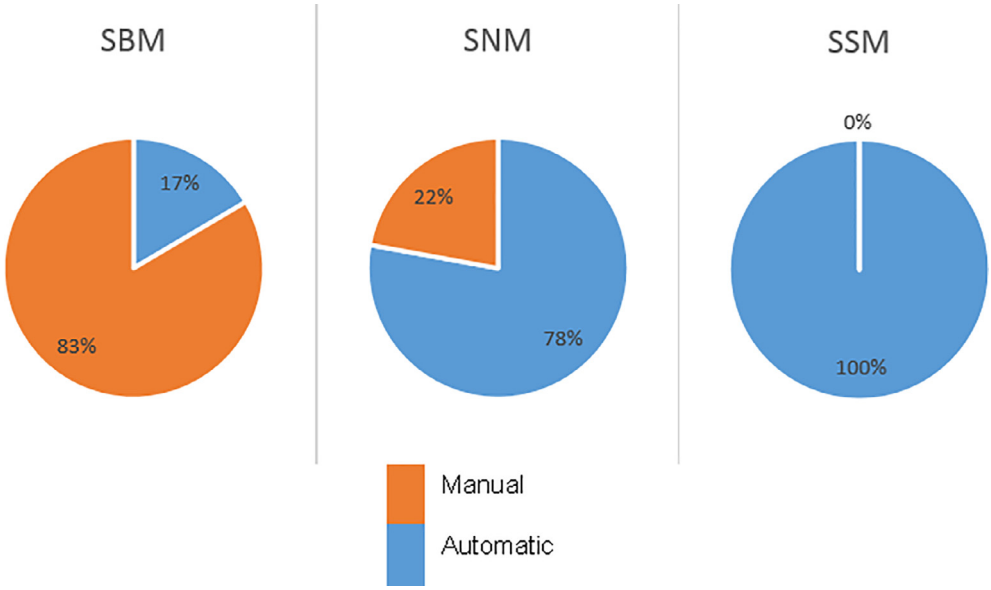

Fig. 6. The comparison of execution time of the tested methods over the *Cora1*.

an expert in the SBM method on the one hand, and expert's definition of the rules, for the classification of record pairs in the SNM method, on the other.

The SSM method gave the best results in testing of the *Dataset1* from the aspect of efficiency since it has the shortest execution time period. It is particularly important to emphasize that in the SSM method all testing phases were executed as *Automatic*, i.e. by the computer without human expert participation. From the aspect of indicators of relative time spent per identified *tp* record pair (Time per *tp*) (Hadzic et al., 2015), it can be noticed that the SSM method is 110 times faster per an identified *tp* record pair compered to the SNM method, and 80 times faster compared to the SBM method.

Table 8 presents the summary and compares the three methods' testing results for the *Cora1*. From the aspect of effectiveness, the best results over the mentioned data set were obtained by the Semantic-Syntactic Method. *Cora1* data set presents a cluster of mutually weakly-paired records which caused weaker result for certain methods from the aspect of effectiveness. From the aspect of precision, all three methods gave high-quality results. Only the Sorted Neighborhood Method does not have the maximum result for precision (*Precision* = 0.999), i.e. it has two record pairs wrongly identified as duplicates (*fp* = 2). However, there is a greater difference in values for *Recall* and *F-score* parameters based on the methods shown in Table 8 and Fig. 5. According to these two parameters, the Semantic-Syntactic Method gave by far the best results, followed by the Sorted Neighborhood Method and the Standard Blocking Method with the F-score 0.401.

The Standard Blocking Method is time-consuming in Manual execution phases, i.e. there is human participation in these execution phases. The reason for this is a large number of record pairs classified as possibly matched (A_2), so the additional human expertise was necessary.

The reason why in the Sorted Neighborhood Method the time spent in Manual execution phase was 22% is because Automatic execution phase, where the comparison of records through the defined rules and identification of a large number of records through certain passes are done, was significantly longer. Also, at the end of the method execution, a significant amount of time was spent on the transitive closure over the already identified *tp* record pairs. The Semantic-Syntactic Method in testing of *Cora1* data set and from the aspect of effectiveness provided the best results Fig. 6), i.e. it has the shortest execution time.

6. Conclusion

According to the presented results, it can be concluded that the Semantic-Syntactic Method is the most effective one for both the tested sets of data. The greatest effectiveness of the Semantic-Syntactic Method is the direct consequence of the following:

- Measuring the similarity between the values of attributes with replaceable values.
- Context key introduction and the separation of different records which contributed to the increase of record comparisons that are duplicate records.
- A better quality syntactic key compared to the blocking and sorting keys, due to the separation of pattern from the defined attribute values, extending the possibility of having the matched records one next to another.
- The selection of a particular number of attributes having a better quality values and with lower number of blank values, contributed to a better quality in calculation of similarities between the record pairs.
- Dynamic record pair identification for comparison in terms of determining which record pair will be compared next according to the previous comparisons.

The execution time and the types of execution in testing methods are always combined with the results related to a number of accurately identified duplicate records (total time (ms)/total number of *tp*). On the basis of previous parameters it can be concluded that the Semantic-Syntactic Method is conceivably most efficient one for both the tested sets of data. This conclusion that the Semantic-Syntactic Method has the greatest efficiency is confirmed by the following:

- The shortest total time for the execution of the method,
- Complete exclusion of manual work (expertise) from the process of method execution,
- The shortest time period spent per one accurately identified duplicate record.

Future research in this area should focus on the ways and methods of merging duplicate records into one record. This is especially interesting from the aspect of record's multiplicity. Specifically, in the data set *Cora1* it can be seen that besides doubled records there are also multiplied records (two or more of the fuzzy same records). Although, there is a certain research work and developed methods related to merging more fuzzy records into one, this process of improved data quality is still very challenging for the researchers.

Appendix A. Appendix

(See Tables 4, 5, 7, 8)

Table 4
The SSM method – *Dataset1* – Passage 1, Passage 2 and Passage 3.

Phase	Results		
	Passage 1	Passage 2	Passage 3
1.Syntactic keys creation	The *Dataset1* sorted by the Context key and Syntactic key 1.	The *Dataset1* sorted by the Context key and Syntactic key 2.	The *Dataset1* sorted by the Context key and Syntactic key 3.
2.1. m_i calculation	Iterations for m_i: 2 $m_0 = 0.984$ $m_1 = 0.854$ $m_2 = 0.820$ $m_3 = 0.885$ $m_4 = 0.783$ $m_5 = 0.910$ $m_6 = 0.996$ $m_7 = 0.962$ $m_8 = 0.910$ $m_9 = 0.962$	Iterations for m_i: 6 $m_0 = 0.837$ $m_1 = 0.774$ $m_2 = 0.804$ $m_3 = 0.887$ $m_4 = 0.780$ $m_5 = 0.986$ $m_6 = 0.991$ $m_7 = 0.986$ $m_8 = 0.909$ $m_9 = 0.953$	Iterations for m_i: 5 $m_0 = 0.8339$ $m_1 = 0.750$ $m_2 = 0.808$ $m_3 = 0.888$ $m_4 = 0.776$ $m_5 = 0.909$ $m_6 = 1$ $m_7 = 0.967$ $m_8 = 0.884$ $m_9 = 0.963$
2.2.Selection of the attributes for $m_i > T_m(0.3)$	All attributes are selected for further analysis ($m_i > 0.3$)	All attributes are selected for further analysis ($m_i > 0.3$)	All attributes are selected for further analysis ($m_i > 0.3$)
2.3. Creating the rules for additional expertise	Rule for Passage 1: *same* (*postcode*) Λ *verysimilar* (*state*) Λ *similar*(*soc_sed_id*)	Rule for Passage 2: *same* (*postcode*) Λ *same* (*soc_sec_id*) Λ *verysimilar* (*date_of_birth*)	Rule for Passage 3: *same (state)* Λ *verysimilar* (*suburb*) Λ *verysimilar* (*address_1*)
3. u_i calculation	$u_0 = 0.483$ $u_1 = 0.279$ $u_2 = 0.279$ $u_3 = 0.289$ $u_4 = 0.254$ $u_5 = 0.295$ $u_6 = 0.333$ $u_7 = 0.$ $u_8 = 0.337$ $u_9 = 0.313$	$u_0 = 0.291$ $u_1 = 0.268$ $u_2 = 0.279$ $u_3 = 0.319$ $u_4 = 0.269$ $u_5 = 0.384$ $u_6 = 0.362$ $u_7 = 0.495$ $u_8 = 0.350$ $u_9 = 0.329$	$u_0 = 0.186$ $u_1 = 0.167$ $u_2 = 0.196$ $u_3 = 0.204$ $u_4 = 0.171$ $u_5 = 0.204$ $u_6 = 0.437$ $u_7 = 0.291$ $u_8 = 0.278$ $u_9 = 0.215$
4.Thresholds similarities	$T_\lambda = -1.90140$ $T_\mu = -0.06015$	$T_\lambda = -0.85717$ $T_\mu = 0.02149$	$T_\lambda = -0.00868$ $T_\mu = 0.00946$
5.Classification ($k = 1$)	Pairs of records: 1001Duplicates: 382 (382)[3] Transitive closure[4]: 0 Unknown records: 2	Pairs of records: 1326Duplicates: 430 (102) Transitive closure: 0Unknown records: 2 (2)	Pairs of records: 1413Duplicates: 425 (11) Transitive closure: 0Unknown records: 0 (0)
Execution	Time (ms): 706.6538 Type: Automatic	Time (ms): 892.1471 Type: Automatic	Time (ms): 894.6692 Type: Automatic

[3] The number of new duplicates as compared to the previous passage is given in parentheses.
[4] Transitional closure performed within a cluster (Fig. 2 – pairs od records under b1).

Table 5
The SSM method – *Cora1* – Passage 1, Passage 2 and Passage 3.

Phase	Results		
	Passage 1	Passage 2	Passage 3
1.Syntactic keys creation	The *Cora1* sorted by the Context key and Syntactic key 1.	The *Cora1* sorted by the Context key and Syntactic key 2.	The *Cora1* sorted by the Context key and Syntactic key 3.
2.1. m_i calculation	Iterations for m_i: 17 $m_0 = 0.955$ $m_1 = 0.211$ $m_2 = 1$ $m_3 = 0.$ 0.044 $m_4 = 0.844$ $m_5 = 0.288$ $m_6 = 0.633$ $m_7 = 0.799$ $m_8 = 0.588$ $m_9 = 0.077$ $m_{10} = 0.044$ $m_{11} = 0.088$ $m_{12} = 1$	Iterations for m_i: 21 $m_0 = 0.873$ $m_1 = 0.339$ $m_2 = 1$ $m_3 = 0$ $m_4 = 1$ $m_5 = 0.058$ $m_6 = 0.252$ $m_7 = 0.941$ $m_8 = 0.912$ $m_9 = 0$ $m_{10} = 0.009$ $m_{11} = 0.058$ $m_{12} = 1$	Iterations for m_i: 18 $m_0 = 0.898$ $m_1 = 0.414$ $m_2 = 1$ $m_3 = 0.060$ $m_4 = 0.858$ $m_5 = 0.100$ $m_6 = 0.162$ $m_7 = 0.919$ $m_8 = 0.898$ $m_9 = 0.010$ $m_{10} = 0.070$ $m_{11} = 0.080$ $m_{12} = 1$
2.2.Selection of the attributes for $m_i > T_m(0.3)$	Author, Title, Venue, Publisher, Year, Pages, Class	Author, Title, Venue, Year, Pages, Class	Author, Volume, Title, Venue, Year, Pages, Class
2.3. Creating the rules for additional expertise	Rule for Passage 1: *same* (*Class*) Λ *verysimilar* (*Venue*) Λ *similar* (*Year*)	Rule for Passage 2: *same* (*Venue*) Λ *same* (*Class*) Λ *verysimilar* (*Year*)	Rule for Passage 3: *same (Class)* Λ *verysimilar* (*Author*) Λ *verysimilar* (*Pages*)
3. u_i calculation	$u_0 = 0.313$ $u_1 = 0.020$ $u_2 = 0.452$ $u_3 = 0.004$ $u_4 = 0.199$ $u_5 = 0.028$ $u_6 = 0.075$	$u_0 = 0.429$ $u_1 = 0.046$ $u_2 = 0.324$ $u_3 = 0.001$ $u_4 = 0.182$ $u_5 = 0.008$	$u_0 = 0.216$ $u_1 = 0.070$ $u_2 = 0.307$ $u_3 = 0.010$ $u_4 = 0.403$ $u_5 = 0.040$ $u_6 = 0.066$
4.Thresholds similarities	$T_\lambda = -$Infinity $T_\mu = 1.32521$	$T_\lambda = -$Infinity $T_\mu = 0.12111$	$T_\lambda = -16.08694$ $T_\mu = 0.01480$
5.Classification ($k = 1$)	Pairs of records: 499Duplicates: 6710 (6710) Transitive closure: 6266 Unknown records: 0	Pairs of records: 1112 Duplicates: 633(15) Transitive closure: 2Unknown records: 0 (0)	Pairs of records: 1945 Duplicates: 1471(28) Transitive closure: 0 Unknown records: 70(1))
Execution	Time (ms): 706.6538 Type: Automatic	Time (ms): 892.1471 Type: Automatic	Time (ms): 894.6692 Type: Automatic

Table 6
SSM – CORA(*Cora1*) – The pairs of records for additional expertise.

Vector	rec_id1	rec_id2	Weight	Classification	Passage
1,010,001	53	62	−1.93998686265674	A_2	3

Table 7
Comparison of the testing results for the *Dataset1*.

Method	SBM	SNM	SSM
Context key	X	X	Yes
Passages	3	3	3
Duplicates through passages	458	474	497
Duplicates – additional expertise	20	X	0
Duplicates – transitive closure	0	0	0
Total duplicates	478	474	497
tp (true positive)	478	474	497
fp (false positive)	0	0	0
fn (false negative)	22	26	3
Precision	1	1	1
Recall	0.956	0.9478957915	0.9939879759
F-score	0.9775051124	0.9732510288	0.9969849246
Time – *Automatic* (ms)	3347.831	3526.54	3638.77
(percent of total time)	(1.19%)	(0.92%)	(100.00%)
Time – *Manual* (ms)	278836.60	378611.27	0 (0.00)
(percent of total time)	(98.81%)	(99.08%)	
The total time (ms)	282184.430	382137.80	3638.77
Time per *tp* (ms)	590.344	806.1978943	7.3214688128

Table 8
Comparison of the testing results for the *Cora1*.

Method	SBM	SNM	SSM
Context key	X	X	Yes
Passages	3	3	3
Duplicates through passages	1434	2307	6753
Duplicates – additional expertise	56	X	1
Duplicates – transitive closure	474	2258	198
Total duplicates	1964	4563	6952
tp (true positive)	1964	4561	6952
fp (false pozitive)	0	2	0
fn (false negative)	5857	3258	869
Precision	1	0.999	1
Recall	0.251	0.583	0.888
F-score	0.401	0.736	0.941
Time – *Automatic* (ms)	91935.5605	289952.37	95921.84
(percent of total time)	16.50%	77.87%	100.00%
Time – *Manual* (ms) (percent of total time)	465205.06 83.50%	82399.99 22.13%	0 0.00%
The total time (ms)	557140.6156	372352.36	95921.84
Time per *tp* (ms)	283.6765	81.60253417	13.7977335

References

Branchman, R.J., Anand, T., 1996. The process of knowledge discovery in databases: A human-centered approach. In Advances in Knowledge Discovery and Data Mining, chapter 2, AAAI/MIT Press, pp. 97–158.

Christen, P., 2012. 2012, Book: "Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection". Springer.

Christen, P., 2012. A survey of indexing techniques for scalable record linkage and deduplication. IEEE Trans. Knowledge Data Eng. 24 (9), 1537–1555. https://doi.org/10.1109/TKDE.2011.127.

Draisbach, U., Naumann, F., 2011. A Generalization of Blocking and Windowing Algorithms for Duplicate Detection. Data and Knowledge Engineering (ICDKE), International Conference; pp. 18–24.

Draisbach, U., Naumann, F., Szott, S., & Wonneberg, O. (2012). Adaptive windows for duplicate detection. Proceedings – International Conference on Data Engineering, 1073–1083. https://doi.org/10.1109/ICDE.2012.20

Elmagarmid, A.K., Ipeirotis, P.G., Verykios, V.S., 2007. Duplicate Record Detection: A Survey. IEEE Trans. Knowledge Data Eng. 19 (1), 1–16.

Hadzic, D., Sarajlic, N., Malkic, J,. 2016. Different similarity measures to identify duplicate records in relational databases, 24th Telecommunications Forum (TELFOR), IEEE Conference Publications, pp. 1 – 4 DOI: 10.1109/TELFOR.2016.7818899.

Hadzic, D., Sarajlic, N., Malkic, J., 2015. Comparation of Partitioning Methods for Identification of Duplicate Records in Databases, BH elektrotehnika, 9, pp. 26–35. (article in Bosnian with an abstract in English – http://bhkcigre.ba/bhs/Izdanja-ba/casopis-elektrotehnika/broj-9 the whole article can be seen at http://bih5.net/nir/cpmiddb_bhelektrotehnika2015.pdf).

Hernández, M.A., Stolfo, S.J., 1998. Real-world data is dirty: data cleansing and the merge/purge problem. Data Min. Knowledge Discov., 9–37

http://lingvo.info/en/babylon/semantics, Babylon, Semantics, lexicology "The meaning behind words".

Fellegi, I.P., Sunter, A.B., Dec. 1969. A Theory for Record Linkage. J. Am. Statistical Assoc. 64 (328), 1183–1210.

Hylton, J.A., 1996. Identifying and Merging Related Bibliographic Records Master's thesis. Department of Electrical Engineering and Computer Science, MIT.

Jaro, M.A., 1989. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, Florida. J. Am. Statis. Assoc., 414–420

Ripon, K.S.N., Rahman, A., Rahaman, G.M.A., December 2010. A domain-independent data cleaning algorithm for detecting similar-duplicates. J. Comput. 5 (12), 1800–1809. https://doi.org/10.4304/jcp.5.12.1800-1809.

L. Gu and R. Baxter. Adaptive filtering for efficient record linkage. In Proceedings of the Fourth SIAM International Conference on Data Mining, 2004.

Lee, M., Ling, T., 1999. Cleansing data for mining and warehousing. 10th International Conference DEXA '99, pp. 751–760.

M. A. Hernandez and S. J. Stolfo, "The merge/purge problem for large databases," in Proceedings of the ACM International Conference on Management of Data (SIGMOD), 1995, pp. 127–138.

Weis, M., Naumann, F., Jehle, U., Lufter, J., and Schuster, H., "Industry-scale duplicate detection," In Proc. International Conference on Very Large Databases, Auckland, NZ, vol. 1, no. 2, pp. 1253–1264, 2008.

Maletic, J.I., Marcus, A., 2000. Data cleansing: Beyond integrity checking. In Proceedings of the Conference on Information Quality (IQ '2000), pp. 200–209.

Maydanchik, A., 1999. Challenges of efficient data cleansing. In DM Review.

Naumann, F., Herschel, M., 2010. An Introduction to Duplicate Detection, Morgan&Claypool Publishers.

Newcombe, H.B., Kennedy, J.M., Axford, S.J., James, A.P., 1959. Automatic linkage of vital records. Science, 954–959.

Tayi, G.K., Ballou, D.P., 1998. Examining data quality. Commun. ACM 41, 54–57.

Yan, S., Lee, D., Kan, M.Y., Giles, C.L., 2007. Adaptive sorted neighborhood methods for eficient record linkage. In Proceedings of the Joint Conference on Digital Libraries (JCDL), Vancouver, Canada, pp. 185–194.