Token-based method of blocking records for large data warehouse

Jebamalar Tamilselvi J.* and Saravanan V.
*Department of Computer Application, Karunya University, Coimbatore, 641114, Tamilnadu, India,
jebamalar@karunya.edu, saravanan@karunya.edu

Abstract: Record linkage is a critical problem in duplicate data elimination. It is used to detect and eliminate duplicate data. The elimination of duplicate data will increase the quality of data. Record Linkage problem will take high computational cost because of the large number of record comparisons. The comparison of records is inefficient in large data warehouses. Blocking methods are used to group the records to minimize the number of record comparisons. This paper explains the existing blocking methods and its comparison and discusses the selection of token-based blocking key for record comparisons.

Keywords: Data Warehouse, Record Linkage, Token, Blocking Records, Record Comparisons, Duplicate Data

1 Introduction

Record linkage [RL] is the process of matching records across data sets that refer to the same entity [14]. Theoretically, RL compares all records in the data sets under analysis in order to decide which record belongs to the same individual. In practice, since the size of data sets is usually very large, comparing all the records between them becomes unfeasible. Therefore, RL resorts to blocking methods [5, 6, 15] that try to gather all the records that present a potential resemblance, only applying RL within each block. Typically, blocking methods are based on a common attribute without errors. Clustering is the classification of objects into different groups, or more precisely, the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait - often proximity according to some defined distance measure. Clustering method is also known as Blocking method in data cleaning for duplicate detection. Blocking typically refers to the procedure of subdividing data bases into a set of mutually exclusive subsets (blocks) under the assumption that no matches occur across different blocks. Although blocking can substantially increase the speed of the comparison process, it can also lead to an increased number of false mismatches due to the misuse of blocking key and the size of the window. There are two main goals of blocking. First, the number of candidate matches generated should be small to minimize the number of detailed comparisons in the record linkage step. Second, the candidate set should not leave out any possible true matches, since only record pairs in the candidate set are examined in detail during record linkage. These blocking goals represent a trade off. On the one hand, the goal of record linkage is to find all matching records, but the process also needs to scale. This makes blocking a challenging problem [14]. The main purpose of this paper is to substantially reduce the probability of false mismatches and to increase the running time, by reducing the number of comparisons. This paper addresses existing blocking methods in Record Linkage providing a summary and comparison of the same. Also, this paper introduces the efficient token-based blocking key selection blocking method. The blocking key is important in this blocking method to gather resemblance records.

2 Clustering / Blocking Methods

2.1 Standard Blocking

The Standard Blocking (SB) method cluster records into blocks where they share the identical blocking key. A blocking key is defined to be composed from the record attributes in each data set. For example, the blocking key is generated by taking first four or three characters of a name attribute. A blocking key can also be composed of more than one attribute. The resulting total number of record pair comparisons of the standard blocking is O(n2/b)[1].

2.2 Sorted Neighborhood Method (SNM)

One of the most common duplicate detection methods is the Sorted Neighborhood method. The Sorted Neighborhood method sorts the records based on a sorting key [SK] and then moves a window called Sliding window (SW) of fixed size w sequentially over the sorted records. Records within the window are then paired with each other and included in the candidate record pair list. The use of the window limits the number of possible record pair comparisons for each record to 2w−1 [5], [6]. The resulting total number of record pair comparisons of the sorted neighborhood method is O(wn).

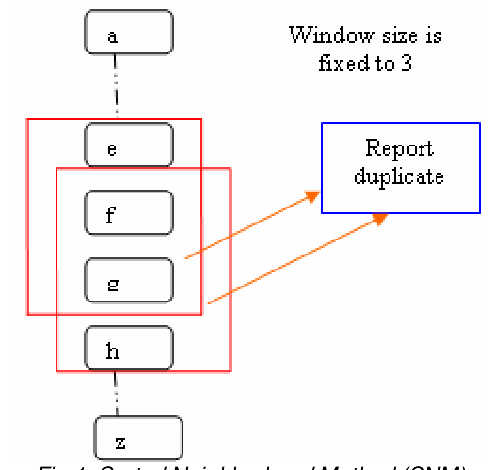

Fig 1: Sorted Neighborhood Method (SNM)

One problem with the sorted neighborhood method is, if a number of records larger than the window size have the same value in a sorting key. Similar to standard blocking, it is advantageous to do several passes (iterations) with different sorting keys and a smaller window size than one pass only with a large window size [7]. The effectiveness of this approach depends on the quality of key chosen to do the sorting [2].

2.3 Multi-pass SNM

The multi-pass sorted neighborhood method involves using multiple passes of the initial sorted neighborhood method. It is executed in several independent runs of SNM, each time using a different key and a relatively small window [4]. In general, no single key will be sufficient to catch all matching records [7]. During each pair, the three steps - create keys, sort data and merge - are performed selecting a different key field for each pass. Once the duplicate records are identified during a pass one of the duplicates is eliminated from the data source. This method provides better results than the SNM method [2].

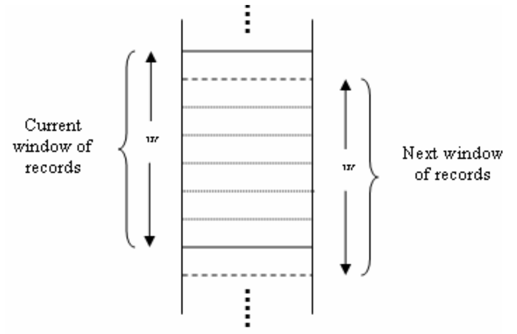

Fig. 2: Sliding Window

2.4 Clustering SNM

Clustering SNM, first partitions the database into independent clusters using a key extracted from the data [3]. Then SNM is applied to each individual cluster independently. There are two phases in Clustering SNM.

1. Cluster data

Scan the records in sequence to extract a key for each record; then using the extracted key, partition the records into independent subsets of data (clusters).

2. SNM

Apply SNM independently on each cluster. The key does not need to be recomputed, the key extracted above can be used for sorting.

2.5 Duplicate Elimination SNM (DE-SNM)

Duplicate Elimination SNM sorts the records on a chosen key and then divides the sorted records into two lists. 1) a duplicate list 2) a non-duplicate list. A small window scan is first performed on the duplicate list to find the list of matched and unmatched records. The list of unmatched records is merged with the original non-duplicate list and a second window scan is performed [4]. Using Sorted Neighborhood method the exact or very closely matching duplicate records with equivalent keys can be identified and it would likely be more efficient to find these records first during the sorting phase [7]. This method removes other duplicate records and retains only one representative member of this set of duplicates by the rule set against other records with different keys. DE- SNM does not contribute much on the improvement of accuracy of SNM. The benefit of DE-SNM is that it runs faster than SNM under the same window size, especially for the databases that are heavily dirty. If the number of records in duplicate list is large, DE-SNM will run faster than SNM [8].

2.6 SNM – Priority Queue

The algorithm scans the database sequentially and determines whether each record is scanned or not and also determines whether a member of the cluster is represented in priority queue [4].This queue contains a fixed number of record sets, which contain similar records that can be paired as candidate duplicates. The sorted list of all records is sequentially scanned and every record Rj is compared with the members Ri in the record set with the highest priority in the queue. If the comparison yields a distance below some threshold T1, the record Rj is included into the set of Ri, if the distance is higher than some threshold T2 this record set is skipped and the comparison is continued with the next highest priority set in the queue. If no set is found at all, a new singleton set is put into the queue. In order to reduce the number of comparisons, the record sets are pruned, i.e. they only contain ”representative” members of the set, in particular, non-representative members are those, which are very similar to other members. However, the authors do not make it clear how to find the two thresholds and how to decide if a record is ”representative” [11].

2.7 K-way Sorting Method

The Sorted Neighborhood Method is not effective where the data source does not contain a primary key field of reference. The k-way sort duplicate detection method is designed to effectively detect duplicates in data sources that do not have any uniquely distinguishing data fields [2, 3]. This K-way sorting method follows the following steps.

The steps are

1. Let k be the number of columns to be used for sorting
2. Select the k-most meaningful combinations of sort keys based on the k selected columns.
3. Assign a record identifier to each record.
4. Sort records based on the selected sort key combination
5. For each sorted set of data, compare adjacent rows within a given window size
6. Draw k graphs for each sort
7. Examine the k graphs collectively. If matches occur between some records or identifier exceeds certain threshold, then it should be mapped into the summation graph.
8. The summation graph should handle transitive closure.

2.8 Bigram Indexing

The Bigram Indexing (BI) method as implemented in the Febrl record linkage system allows for fuzzy blocking. The basic idea is that the blocking key values are converted into a list of bigrams (sub-strings containing two characters) and sub-lists of all possible permutations will be built using a threshold (between 0.0 and 1.0). The resulting bigram lists are sorted and inserted into an inverted index, which will be used to retrieve the corresponding record numbers in a block [1]. The number of sub-lists created for a blocking key value both depends on the length of the value and the threshold. The result of bigram list is smaller blocks will be produced in the inverted index if the value of threshold is low and the sub-list is small with per blocking key value. In the information retrieval field, bigram indexing has been found to be robust to small typographical errors in documents. Like standard blocking, the number of record pair comparisons with two data sets with n records each and b blocks, all containing the same number of records is O(n2 b) [9]. The number of blocks b will be much larger in bigram indexing.

2.9 Canopy Clustering with TF-IDF

Canopy Clustering with TF-IDF (Term Frequency-Inverse Document Frequency) forms blocks of records based on those records placed in the same canopy cluster. A canopy cluster is formed by choosing a record at random from candidate set of records. This canopy clustering method uses the TF-IDF distance metric instead of bigrams [1]. The total number of record pair comparisons resulting from canopy clustering is $O(fn^2/c)$ where n is the total number of records, f is the average number of canopy and c is the number of canopies. The threshold parameter should be set so that f is small and c is large, in order to reduce the amount of computation. However, if f is too small, then the method will not be able to detect typographical errors [1], [9].

2.10 SNM-IN

The main difference between SNM and SNM-IN is as follows: while SNM compares the new record entering the current window with all previous records in the window, SNM-IN will first check whether the new record with previous records are duplicate or non-duplicate with the new record, there is no need to compare them with the new record. Thus, with Lower bound and Upper bound, SNM-IN will reduce a lot of comparisons.

2.11 SNM-INOUT

SNM-IN follows SNM with only one anchor record and SNM-INOUT follows SNM with two anchor records. One anchor record is an inAnchor record and the other is outAnchor record. SNM-INOUT has some changes in Merge phase of SNM method. When a new record enters the current window, it is first compared with the two anchor records, which introduces one more comparison than SNM-IN. However, in SNM-INOUT, for each record in the current window, the chance of determining the new record as duplicate record or not is increased, with produce more chance to reduce comparisons than SNM-IN does. Since the outAnchor is outside the window and the last record in the window will compare with it, SNM-INOUT will obtain a few more duplicate pairs than SNM-IN if both run at the same window size.

2.12 Disjunctive Blocking

Disjunctive blocking selects record pairs that are placed in the same block by at least one blocking predicate from the selected subset of predicates. This strategy can be viewed as selecting pairs for which a disjunction of predicates evaluates to *true*. The blocking function is trained by selecting a subset of blocking predicates [13].

2.13 Disjunctive Normal Form (DNF) Blocking

DNF blocking selects object pairs that satisfy at least one conjunction of blocking predicates from a selected subset of conjunctions. This strategy can be viewed as selecting pairs for which a disjunction of predicate conjunctions evaluates to *true*. The blocking function is trained by constructing the DNF formula over the blocking predicates [13].

2.14 Fuzzy Blocking

In the standard blocking method, the blocking keys are selected from the noisy attributes. The fuzzy blocking method substitutes the blocking key by the results of an OWA (Ordered Weighting Averaging) operator. This fuzzy blocking method follows the following steps:

1. A data in the data set is normalized
2. A Fuzzy quantifier is selected
3. OWA operator with the fuzzy quantifier
4. The OWA result is rounded and the record is blocked using this value as a blocking key.

The fuzzy blocking method has two parameters: The OWA quantifier that decides the way to aggregate, and the rounding method that decides the number of blocks to do [6].

2.15 Adaptive Sorted Neighborhood Method (A-SNM)

Adaptive Sorted Neighborhood Method uses the original SNM approach to slide a fix-sized window n - w + 1 times, to generate n-w+1 number of blocks when applied to n records. Note that these generated blocks are overlapping each other, and fix-sized. However, in the adaptive SNM, the "window" becomes only an interim tool to generate final blocks, and all generated blocks will have different (thus adaptive) sizes [10].

2.15.1 Incrementally Adaptive SNM (IA-SNM)

The basic idea is to measure whether records within a small neighborhood are close/sparse and if there are rooms to grow/shrink in the window, then the window size is increased/decreased dynamically. In order to measure the record distribution within a window, it seems that we need to measure the distances between all the records in the window. If the distance between the first and last record satisfies dist(r1,rw1), where φ is the distance threshold. This distance indicates that records within the current window are close to each other, so there is still room to enlarge the window size to find more potential duplicate records. Otherwise the window should be retrenched [10].

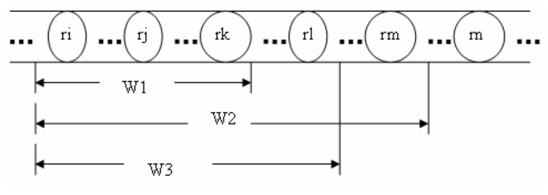

Fig 3: Incrementally-adaptive SNM (IA-SNM)

2.15.2 Accumulatively Adaptive SNM (AA-SNM)

Accumulatively Adaptive SNM is another scheme that can adaptively find various sized blocks. Unlike IA-SNM, which does not explicitly search for boundary pairs to find blocks, the goal of this method is to quickly and accurately find all boundary pairs. As in IA-SNM, AA-SNM measures the distance between the first and the last record in the current window W1. If the distance is less than or equal to a threshold, all the records in window Wm1 can be determined as potential duplicates to each other and should be grouped into the same block. It also suggests that the boundary pair is not in the current window and there are more potential duplicates outside windowWm1. Therefore, the window needs to enlarge the neighborhood of comparison to include more potential duplicates and to search for the boundary pair. This can be achieved by either enlarging the window size, or by moving the window forward and connecting the results of consecutive windows later. The second option is used in AA-SNM. In the retrenchment phase, the binary comparison is done to reduce the time [10].

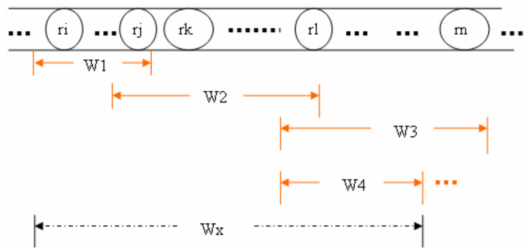

Fig 4: Accumulatively-adaptive SNM (AA-SNM)

3 Comparison of Blocking Methods

In detection methods, the most reliable way is to compare every record with every other record. Obviously this method guarantees that all potential duplicate records are compared and then provides the best accuracy. However, the time complexity of this method is quadratic. It takes N(N-1)/2 comparisons if the database has N records, which will take very long time to execute when N is large. Thus it is only suitable for small databases and is definitely impracticable and infeasible for large databases. Therefore, for large databases, approximate detection algorithms that take far less comparisons (e.g., O(N) comparisons) are required. The window size used in SNM is an important parameter that affects the performance. If the window size is large then accuracy increases slowly but the time increases fast. The duplicate records missed in one pass would be defeated in another pass with different key, so multi-pass increases correctness. Multi-pass SNM can drastically improve the accuracy of the results of only one run of SNM with varying large windows. Particularly, only a small window size is needed for the multi-pass SNM to obtain high accuracy, while no single run with a key can produce comparable accuracy results with a large window. One issue in multi-pass SNM is that it employs transitive closure to increase the number of duplicate records. The transitive closure allows duplicate records to be detected even without being in the same window during an individual window scan. The multi-pass SNM also increases the number of false positives. Priority Queue can save some unnecessary comparisons taken by SNM by computing the transitive closure online. Priority Queue may be faster than SNM but cannot improve the accuracy under the same conditions with SNM. In addition, the performance of priority queue depends on the degree of dirtiness of databases. For clean and slightly dirty databases priority queue does not provide any help. But for dirty databases, priority queue is much faster. The more dirtier the database is, the more time it can save. Like priority queue, DE-SNM can also run faster than SNM for dirty databases, but DE-SNM will decrease the accuracy. Clustering SNM is an alternative method. As the name shows, clustering SNM does one even loser clustering before applying SNM. The clustering SNM is faster than SNM for very large databases but it may decrease the accuracy as well. Further, clustering SNM is suitable for parallel implementation. The bigram indexing needs the inverted index for the bigram sub-lists, where as canopy clustering needs a special index for finding all records that, are within the threshold of a distance measure, which depends on the used distance measure [12]. The k-way sort duplicate detection method is designed to effectively detect duplicates in data sources that do not have any uniquely distinguishing data fields [2, 3]. Disjunctive and DNF blocking can then be performed by iterating over the lists for each block in the inverted indices and returning all pairs of records that co-occur in at least one such list. DNF blocking is more accurate than disjunctive blocking at 100% recall [13]. SNM and SNM-IN obtain exactly the same duplicate result, and SNM-INOUT obtains slightly more duplicate pairs than SNM-IN does. In SNM-IN and SNM-INOUT, the number of comparisons is saved with different window sizes. SNM-INOUT will take more comparisons than SNM-IN because SNM-INOUT uses two anchor records. IA-SNM and AA-SNM are robust to the variation in the distribution of duplicates. These both methods are adaptive to the variance of block sizes. AA-SNM has better performance than IA-SNM. SNM blocking method gives better performance than other methods like shrinking or expanding the window size based on the records. Finally, all the blocking methods are effective in blocking records but the quality of the data, the time taken for the record comparison and false positive rate are different in all the existing blocking methods. The false positive rate is reduced by the dynamic changes of window size and by the usage of an effective blocking key.

4 Token-Based Blocking key

Blocking is a mechanism used in record linkage to reduce the number of pair comparisons. A database (set of records) is divided into smaller blocks by blocking key values. Instead of comparing every possible pair that can be formed by records in the database, one will need to compare those pairs whose records belong to the same block. A blocking key is a pre-defined set of positions. A requirement for a good blocking key is that, the more the possibility that two records are duplicate, the more likely they are in the same block. The automatic selection of good blocking keys is very important to bring duplicate records together. In the existing method the first three or four characters are selected from the field value of single attribute which is having high power discrimination. This way of blocking key formation is not efficient to bring all the duplicate records together. In some other existing method of blocking, the first character of selected attribute field value is selected as blocking key. Therefore, the selection of attribute is very important in the selection of blocking key. These high power discrimination attributes will be selected based on some criteria. This paper proposes attribute selection method for the data cleaning process. There are three criteria that are used to identify relevant attributes for further data cleaning process. The three criterias are: (a) Identifying key attributes, (b) classifying attributes with high distinct value and low missing value and (c) measurement types of the attributes.

a) Identifying key attributes

The key is an attribute or a set of attributes that uniquely identify a specific instance of the table. Every table in the data model must have a primary key whose values uniquely identify instances of the entity. The key may be primary key, candidate key, foreign key or composite key.

b) Classifying distinct and missing values

Missing character values are always same no matter whether it is expressed as one blank, or more than one blanks. Obviously, missing character values are not the smallest strings. Distinct is used to retrieve number of rows that have unique values for each attribute. The distinct value is used to calculate an Identification power of the attribute. So the distinct value is very much important in attribute selection. An Identification power of the attribute (*ipj*) is used to evaluate the discriminating power of record attributes.

$$\text{Identification power of the attribute j (ipj)} = \frac{\text{Number of distinct equivalence classes on the total of record}}{\text{Total number of records}}$$

c) Classifying types of attributes

There are four types of attributes: nominal, ordinal, interval and ratio. Different criteria are given for each attribute type. The value of these measurement types are also considered for the attribute selection. The data cleaning using numeric data will not be effective. Categorical data is efficient for the data cleaning process.

The selected attributes will be used as the blocking key. If the window size is small, then the number of comparisons will be reduced but the false-positive value will be increased. If the window size is large then the number of comparisons will be increased but the false-positive value will be reduced. For this kind of problem, the calculation of distinct and missing value is very important in the selection of blocking key. The numeric blocking key will not be effective in blocking the record for comparison. So, the identification of measurement type and type of the attribute is important in the selection of blocking key.

Token is formed for the selected attribute field value. Different token formation rules are followed for different kind of data. The data may be numeric, alphanumeric or alphabetic. The rules are given in the following steps.

a. Numeric Tokens: This numeric token formation rule is suitable for phone number, social security number, street number, apartment number, and so on. First, it removes the unimportant characters and converts the character to numeric. Finally, groups the number to keep together as one token [16, 17].

b. Alphabetic Tokens: This alphabetic token formation rule is well suited for the names such as contact name, customer name, producer name, book title, and so on. First, it expands the abbreviations and removes the unimportant and stopping characters. Finally, takes the first character from each word, sorts the selected character then groups together as one token [16].

c. Alphanumeric Tokens: This alphanumeric token rule is suited for address, product code and so on. First, it splits alphanumeric into numeric and alphabetic, sorts the divided token and then groups numeric and alphabetic separately. Finally, the tokens in the field are grouped together to get token as one field [16, 17].

Table 1: Formation of Tokens for the address field

Customer Credit ID	Address	Token key
1	7464 South Kingsway, Sterling Heights	7464 SKSH
2	410 Eighth Avenue, DeKalb	410 EAD
3	7429 Arbutus Boulevard, Blacklick	7429 ABB
4	8287 Scott Road, Huntsville	8287 SRH
5	480 Grant Way, San Diego	480 GWSD
6	1984 Sydney Street, Austin	1984 SSA
7	7655 Mayberry Crescent, Eden Prairie	7655 MCE
8	413 Robson Lane, Winchester	413 RLW
9	1842 St. Anne Place, Concord	1842 APC
10	8404 Nelson Lane, Winchester	8404 NLW

This proposed work uses the existing blocking key approaches as well as it uses token based blocking key. The token is formed for each selected attribute field values before blocking the records. Then the blocking key is created from the token field by selecting the first character of each token filed. The sample blocking key formation is listed in Table2. The token based blocking key is efficient to block the records. While blocking the records, the similarity function is used to group the records based on the threshold value. The proposed work will be efficient to reduce the false positive.

Table 2: Token based Blocking key

Customer Credit ID	Contact name key	Customer name key	Address key	Block - Token Key
1	AB	BPT	6938 BSM	AB6
2	BC	ACCT	1867 TLC	BA1
3	AS	CP	8194 PAEP	AC8
4	AS	CRR	4861 SRN	AC4
5	GJ	AABH	7429 ABB	GA7
6	AM	PC	8287 SRH	AP8
7	PR	ACC	1842 APC	PA1
8	A	PC	8287 SRH	AP8
9	BM	CCGS	413 RLW	BC4
10	CC	BCT	8404 NLW	CB8
11	CC	BCT	8404 NLW	CB8
12	CC	CC	7464 SKSH	CC7
13	DH	FJR	1984 SSA	DF1
14	DH	FJR	1984 SSA	DF1
15	CK	LSW	3 802 GCDM	CL3

5. Conclusion

In this paper, existing blocking methods are presented for the blocking records to minimize the time and number of comparison in the record linkage. The selection of the most suitable blocking key (parameter) for the blocking method is addressed in this paper. Dynamically adjusting the blocking key for the blocking method will be effective in record linkage algorithms during the execution time. The blocking key is selected based on the type of the data and usage of the data in the data warehouse. The dynamically adjusting blocking key and token based blocking key as well as the dynamic window size SNM method will be used in this proposed work. Future directions of this work include agents in tuning parameter or everything is set dynamically for the blocking method without human intervention to yield better performance. However, in most real world problems where expert knowledge is hard to obtain, it is helpful to have methods that can automatically choose reasonable parameters for us.

Reference

[1] Rohan Baxter, Peter Christen and Tim Churches (2003) *Record Linkage and Object Consolidation at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington DC.*

[2] Chen Shengxin (2002) *Morgan Kaufmann Publishers Language: ENGLISH. 16x24 Hardback*, 21-26.

[3] Feekin A. and Chen Z. (2000) *Proceeding ACM SAC Conference*, 323-327.

[4] Li Zhao, Sung Sam Yuan, Sun Peng, Tok Wang Ling (2002) *Proceedings of the 13th International Conference on Database and Expert Systems Applications, Publisher-Springer-Verlag,* 2453, 484 – 493.

[5] Mauricio A. H. and Stolfo J. S. (1998*) Data Mining and Knowledge Discovery*, 9-27.

[6] Jordi Nin and Vicen Torra C. *IIIA, Artificial Intelligence Research Institute. CSIC, Spanish National Research Council. Campus UAB.*

[7] Monica Scannapieco, Massimo Mecella, Tiziana Catarci, Cinzia Cappiello, Barbara Pernici, Francesca Mazzoleni (2003) *Data Quality in Cooperative Information Systems, DaQuinCIS, Gennaio,*10.

[8] Sung S. Y., Li Z. and Ling T.W. (2003) *Clustering and Information Retrieval*, 227-260.

[9] Markus Döring, Andreas Müller. (2007) *Botanischer Garten und Botanisches Museum Berlin-Dahlem, Freie Universität Berlin.*

[10] Su Yan, Dongwon Lee, Min-Yen Kan and Lee Giles C. (2007) *JCDL*, 185-194.

[11] Alvaro Monge E. and Charles Elkan (1996) *Knowledge Discovery and Data Mining*, 267-270.

[12] Partrick Lehti (2006) *Lecture Notes in Computer Science,* 4244, 136-164.

[13] Bilenko M., Kamath B., Mooney R.J. (2006) *ICDM Sixth International Conference on Data Mining*, 87 – 96.

[14] Michelson M. and Knoblock C.A. (2006) *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06), Boston, Massachusetts,* 440-445*.*

[15] Jaro M. A. (1989) *Journal of the American Statistical Society*, 84(406), 414-420.

[16] Ohanekwu T.E. and Ezeife C.I. (2003) *Proceedings of the International Workshop on Data Quality in Cooperative Information Systems, 9th International Conference on Database Theory (ICDT),* 21-26.

[17] Ezeife C.I. and Timothy Ohanekwu E. (2005) *The International Journal of Data Warehousing and Mining (IJDWM),* 1(2), 1-22.