A Mixed Similarity Measure in Near-Linear Computational Complexity for Distance-Based Methods

Ngoc Binh Nguyen * and Tu Bao Ho

Graduate School of Knowledge Science
Japan Advanced Institute of Science and Technology
Tatsunokuchi, Ishikawa, 923-1292 JAPAN
{binh,bao}@jaist.ac.jp

Abstract. Many methods of knowledge discovery and data mining are distance-based such as nearest neighbor classification or clustering where similarity measures between objects play an essential role. While real-world databases are often heterogeneous with mixed numeric and symbolic attributes, most available similarity measures can only be applied to either symbolic or numeric data. In such cases, data mining methods often require transforming numeric data into symbolic ones by discretization techniques. Mixed similarity measures (MSMs) without discretization of numeric values are desirable alternatives for objects with mixed symbolic and numeric data. However, the time and space complexities of computing available MSMs are often very high that make MSMs not applicable to large datasets. In the framework of Goodall's MSM inspired by biological taxonomy, computing methods have been done but their time and space complexities so far are at least $O(n^2 \log n^2)$ and $O(n^2)$, respectively. In this work, we propose a new and efficient method for computing this MSM with $O(n \log n)$ time and $O(n)$ space complexities. We demonstrate experimentally the applicability of new method to large datasets and suggest meta-knowledge on the use of this MSM. Practically, the experimental results show that only the near-linear time and space MSM could be applicable to mining large heterogeneous datasets.
Keywords: mixed similarity measure (MSM), computational complexity, very large databases, distance-based methods.

1 Introduction

Traditionally, distance-based methods can be applied to datasets which contain either symbolic or numeric data as most similarity measures are available for homogeneous data. However, real-world data are often heterogeneous with mixed numeric and symbolic attributes. In such cases, most distance-based methods require transforming numeric data into symbolic ones by discretization techniques.

* Currently with the Faculty of Information Technology, Hanoi University of Technology. Dai Co Viet, Hai Ba Trung, Hanoi, VIETNAM.

D.A. Zighed, J. Komorowski, and J. Żytkow (Eds.): PKDD 2000, LNAI 1910, pp. 211–220, 2000.

Discretization however may cause a loss of information and a difficulty in interpreting results. There have been an interest in finding *mixed similarity measures* (MSMs) for processing mixed symbolic and numeric data [2], [4], [5], [7]. In [2], the authors use a Cartesian join operator whenever a mutual pair of symbolic objects is selected for agglomeration based on minimum dissimilarity. In [5], the author extended the Minkowski metric on multidimensional space containing mixed data. Thought having good properties the authors do not provide any experiment on large real-world data or an analysis of computational complexity of their methods. In [1], Goodall proposed an MSM for biological taxonomy. This MSM provides a uniform framework for processing mixed numeric and symbolic data. However, in this framework the computation for numeric attributes may take the time of $O(n^4)$ where n is the maximum number of unique values met in the database for any numeric attributes. Recently, in [7] the authors introduced a computing method for Goodall's MSM and showed that their computing method (with direct implementation of the MSM) is more efficient than the original one. However, with computation time of $O(n^2 \log n^2)$ and the space of $O(n^2)$, this computing method for MSM has still not be able to be applied to large datasets. We propose a new method for calculating the same MSM of Goodall but with time $O(n \log n)$ and linear space $O(n)$. We show experimentally the applicability of the new method to large datasets. The efficiency of the new computing method makes the MSM applicable to many KDD distance-based methods. Section 2 of this paper summaries the basics of computation for Goodall's MSM. Section 3 provides a computation complexity analysis for MSM as the basics of our computing method, and the description of a linear-time algorithm. Section 4 reports experimental results on efficiency of the proposed method.

2 Basics of Mixed Similarity Measure Computation

The computation of the mixed similarity measure in [1] consists of two phases: (1) computation of the similarity in respect of single numeric and symbolic attributes, and (2) combination of the obtained similarities into a unique similarity value. The key idea of similarity computation is *uncommon attribute values make greater contributions to the overall similarity between two objects.* For any particular attribute, similarity will be defined in such a way that it generates an ordering relationship between pairs of values so that, for two pairs of values (a, b) and (c, d), either (a, b) are more similar than (c, d), or the reverse is true, or the two pairs are equally similar. The definition of similarity depends on the type of attribute in question – whether metrical (numeric) or non-metrical (symbolic). Once all pairs of values for an attribute have been ordered in respect of similarity, *the similarity for each pair of instances is defined as the complement of the probability that a random pair of instances will have a similarity equal to, or greater than, the pair in question.*

Similarity between symbolic values. Pairs of differing values are all regarded as equally dissimilar, but pairs of values which agree are ordered according to the rule: agreement between two instances in possessing an uncommon value

of the attribute is considered as indicating closer similarity than agreement in possessing a commoner value. The possible pairs of values having been ordered in this way, their probabilities are then calculated, and the measure of similarity for any pair is then the complement of the sum of the probabilities for all pairs of values equal or greater in similarity. In practical situation, of course, the p_i will not be known but need to be estimated from the sample of m instances.
Proposition 1 *From a sample of m instances, if f_i is the number of instances in m which have V_i as the value of the attribute in question, the appropriate estimate of similarity S_{ij} between two symbolic values can be*

$$\hat{S}_{ij} = \begin{cases} 1 - \sum_{f_k \le f_i} \frac{f_k(f_k-1)}{m(m-1)}, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases} \tag{1}$$

Similarity between numeric values. Let an attribute take a set of ordered numeric values $V_1 < V_2 < \cdots < V_n$. The numeric values as metric data: they have the order and the difference between any two values. Pairs with identical values are more similar than those which different values, those with a small difference are more similar than those with a larger difference. It appears desirable to take into account the size of the groups encompassed by the two values. Again, once these ordering relations have been established, the degree of similarity is expressed by the complement of the probability of the observed pair or any more similar. Given a pair of numeric values (V_i, V_j), the probability P_{ij} that a random pair of values will be as similar as or more similar than (V_i, V_j) is defined by

$$P_{ij} = \sum_{k \in Q_{ij}} \left(p_k^2 + 2 \sum_{t \in T_k} p_k p_t \right) \tag{2}$$

$$T_k = \left\{ t \mid \left[(|V_t - V_k| < |V_j - V_i|) \vee \left[(|V_t - V_k| = |V_j - V_i|) \wedge \left(\sum_{u=i}^{j} p_u \ge \sum_{u=k}^{t} p_u \right) \right] \right] \wedge [k < t \le n] \right\} \tag{3}$$

$$Q_{ij} = \left\{ k \mid [(i \neq j) \vee (p_k \le p_i)] \wedge [1 \le k \le n] \right\} \tag{4}$$

Proposition 2 *The similarity S_{ij} between two numeric values can be estimated from sample data by*

$$\hat{S}_{ij} = 1 - \hat{P}_{ij} = 1 - \frac{1}{m(m-1)} \sum_{f_k \le f_i} \left(f_k(f_k - 1) + 2 \sum_{t \in T_k} f_k f_t \right) \tag{5}$$

where $\hat{P}_{ij}$ is the estimation of the probability P_{ij} by frequencies. **Combining similarities of numeric and symbolic values**. Goodall in [1], Li and Biswas in [7] have described a method to combine similarities in respect of different attributes by using Fisher's transformation and Lancaster's transformation [6]

as follows. Denote $(P_{ij})_k$ the probability P_{ij} that a random pair of values at attribute k will be as similar as, or more similar than, the pair (V_i, V_j) at attribute k, then such a probability for all t_c numeric attributes is combined by Fisher's χ^2 transformation:

$$(\chi_c)^2_{ij} = -2\sum_{k=1}^{t_c} \ln((P_{ij})_k) \tag{6}$$

where t_c is the number of numeric attributes in the data. Similarly, the probabilities from symbolic attributes are combined using Lancaster's *mean value* χ^2 transformation [6]:

$$(\chi_d)^2_{ij} = 2\sum_{k=1}^{t_d} \left(1 - \frac{(P_{ij})_k \ln(P_{ij})_k - (P_{ij})'_k \ln(P_{ij})'_k}{(P_{ij})_k - (P_{ij})'_k}\right) \tag{7}$$

where t_d is the number of symbolic attributes in the data, $(P_{ij})_k$ is the probability for symbolic attribute value pair $((V_i)_k, (V_j)_k)$ at attribute k, $(P_{ij})'_k$ is the next smaller probability in the symbolic set. The significance value of this χ^2 distribution can be looked up in standard tables or approximated from the expression:

$$P_{ij} = e^{-\frac{\chi^2_{ij}}{2}} \sum_{k=0}^{(t_d+t_c-1)} \frac{(\frac{1}{2}\chi^2_{ij})^k}{k!} \tag{8}$$

where $\chi^2_{ij} = (\chi_c)^2_{ij} + (\chi_d)^2_{ij}$. The overall similarity representing the set of $(t_c + t_d)$ independent similarity measures is $S_{ij} = 1 - P_{ij}$. The similarity measures obtained from individual attributes are combined to give the overall similarity between pairs of objects.

3 An MSM in Linear Time and Space Computation

3.1 New Efficient Computation

Because computing the similarities for symbolic features requires the $O(n)$ time complexity, therefore our efforts are to speed up computing the similarities for numeric features. **Proposition 3** *The following properties of P_{ij} hold regarding $d_{ij} = |V_j - V_i|$:*

1. *$P_{ij} = 1$ iff $d_{ij} = |V_n - V_1|$ (i.e. $\{i, j\} = \{1, n\}$)*
2. *$d_{ij} < d_{kl} \Longrightarrow P_{ij} < P_{kl}$*
3. *$0 < P_{ij} \leq 1$ for $\forall i, j$; but $\hat{P}_{ij} = 0$ iff $(i = j) \wedge (f_i = 1)$*
4. *$P_{ij} > \sum_{k=1}^{n} p_k^2$ for $\forall i \neq j$*
5. *$T_k = \phi$ when $i = j$ in Eqs.(3) and (4)*

Observing these properties, the essential idea of our method is to compute the quantity $\hat{\overline{P}}_{ij}$ as an estimate of $\overline{P}_{ij} = 1 - P_{ij}$ instead of computing $\hat{P}_{ij}$. The following sequential analyses will lead us to the key result formulated in Proposition 4. Note that T_k is a set of indices t $(k < t \leq n)$ such that either

$$|V_t - V_k| < |V_j - V_i| \tag{9}$$

or

$$(|V_t - V_k| = |V_j - V_i|) \wedge (\sum_{u=i}^{j} p_u \geq \sum_{u=k}^{t} p_u) \tag{10}$$

Thus, $\overline{P}_{ij}$ is related to the complement of Eq.(9) and Eq.(10), that is

$$|V_t - V_k| > |V_j - V_i| \tag{11}$$

and

$$(|V_t - V_k| = |V_j - V_i|) \wedge (\sum_{u=i}^{j} p_u < \sum_{u=k}^{t} p_u) \tag{12}$$

In Eq.(3) when $i = j$ we have $T_k = \phi$. Hence

$$P_{ij} = \sum_{k \in \{k | p_k \leq p_i\}} p_k^2, \quad i = j \tag{13}$$

and in this case, the computation of P_{ii} takes the $O(n)$ time. From the definition $\overline{P}_{ij} = 1 - P_{ij}$, in Eq.(4) when $i \neq j$: $Q = \{1, 2, ..., n\}$. Using Eq.(2) we have

$$\overline{P}_{ij} = 2 \sum_{k=1}^{n} \left(p_k \sum_{t \in \overline{T}_k} p_t \right), \ i \neq j \tag{14}$$

$$\overline{T}_k = \left\{ t \mid \left[(|V_t - V_k| > |V_j - V_i|) \vee \left[(|V_t - V_k| = |V_j - V_i|) \wedge \left(\sum_{u=i}^{j} p_u < \sum_{u=k}^{t} p_u \right) \right] \right] \wedge [k < t \leq n] \right\} \tag{15}$$

We can write $\overline{T}_k$ i a simpler form:

$$\overline{T}_k = \{t \mid t_k \leq t \leq n\} \ \cup \ T_k^0 \tag{16}$$

where t_k is the smallest t satisfying Eq.(11), and T_k^0 contains values t satisfying Eq.(12), if any. Note that if there exists $t \in T_k^0$, then t is equal to $t_k \pm 1$. To show the correctness of computing $\overline{P}_{ij}$ in Eq.(14), using Eq.(3) and Eq.(15) to get

$$T_k \cup \overline{T}_k = \{t \mid k < t \leq n\} \tag{17}$$

then summing Eq.(2) and Eq.(14), for $i \neq j$, we have

$$P_{ij} + \overline{P}_{ij} = \sum_{k=1}^{n} \left(p_k^2 \ + \ 2 \sum_{t>k} p_k p_t \right) = \left(\sum_{k=1}^{n} p_k \right)^2 = 1^2 = 1 \tag{18}$$

Using Eq.(14) and Eq.(16) together, we can rewrite $\overline{P}_{ij}$ in the form:

$$\overline{P}_{ij} = 2 \sum_{k=1}^{n} \left(p_k \sum_{t=t_k - |T_k^0|}^{n} p_t \right), \ i \neq j \tag{19}$$

or

$$\overline{P}_{ij} = 2\sum_{k=1}^{n}\left(p_k(1 - \sum_{t=1}^{t_k-|T_k^0|-1} p_t)\right) = 2\sum_{k=1}^{n}(p_k(1-s_k)),\ i \neq j \qquad (20)$$

where

$$s_k = \sum_{t=1}^{t_k-|T_k^0|-1} p_t \qquad (21)$$

Proposition 4 *The probability P_{ij} in Eq.(2) can be estimated by $\hat{P}_{ij} = 1 - \hat{\overline{P}}_{ij}$ where*

$$\hat{\overline{P}}_{ij} = \frac{2}{m(m-1)}\sum_{k=1}^{n}(f_k(1-\hat{s}_k)),\ i \neq j\ ,\ \hat{s}_k = \sum_{t=1}^{t_k-|T_k^0|-1} f_t \qquad (22)$$

For given i, j at step k, all t_k, T_k^0, and s_k could be computed in the $O(1)$ time by using the accumulated information from the previous steps. Therefore, computing $\hat{\overline{P}}_{ij}$ by Eq.(22) takes the $O(n)$ time complexity. In the next subsection we propose such an algorithm to estimate P_{ij} via computing $\hat{\overline{P}}_{ij}$ by Eq.(22). In fact, by the definition of S_{ij}, we have $\overline{P}_{ij} = 1 - P_{ij} = S_{ij}$, or $\hat{\overline{P}}_{ij} = 1 - \hat{P}_{ij} = \hat{S}_{ij}$. In sum, we have the following result:

The probability $\hat{P}_{ij}$ in Eq.(5), used to be computed with the $O(n^2 \log n^2)$ time and $O(n^2)$ space complexities, could be computed by the complement $\hat{\overline{P}}_{ij}$ of its estimation $\hat{P}_{ij}$ in Eq.(22) with $O(n)$ time and $O(n)$ space complexities.

Example. Consider a set of $n = 5$ values with occurrence probabilities as follows

i	1	2	3	4	5
V_i	5.5	6.0	7.5	9.0	10.5
p_i	0.1	0.2	0.4	0.2	0.1

Let take (V_3, V_5), then $d_{3,5} = |10.5 - 7.5| = 3.0$. Consider step by step $k = 1 : t_k = 4, T_k^0 = \phi, s_k = 0.7, \overline{P}_{3,5} = 2 * 0.1 * (1.0 - 0.7) = 0.06$,
$k = 2 : t_k = 5, T_k^0 = \{4\}, s_k = 0.9, \overline{P}_{3,5} = 0.06 + 2 * 0.2 * (1 - 0.9) + 2 * 0.2 * 0.2 = 0.18$,
$k = 3 : t_k = 6, T_k^0 = \phi, s_k = 1.0, \overline{P}_{3,5} = 0.18 + 2 * 0.1 * (1.0 - 1.0) = 0.18$,
$k = 4$ and 5 are not considered because $s_k = 1.0$. Hence, $P_{3,5} = 1 - \overline{P}_{3,5} = \mathbf{0.82}$.
For direct computation: $T_1 = \{2, 3\}$; $T_2 = \{3\}$; $T_3 = \{4, 5\}$; $T_4 = \{5\}$; $T_5 = \phi$, $Q = \{1, 2, 3, 4, 5\}$, so $P_{3,5} = 0.1^2 + 2 * 0.1 * (0.2 + 0.4) + 0.2^2 + 2 * 0.2 * 0.4 + 0.4^2 + 2 * 0.4 * (0.2 + 0.1) + 0.2^2 + 2 * 0.2 * 0.1 + 0.1^2 = \mathbf{0.82}$, exactly as $1 - \overline{P}_{3,5}$. □

3.2 Algorithm

Fig.1 describes an algorithm for computing P_{ij} via $\overline{P}_{ij}$. The inputs to the algorithm are: $\{V_i\}_{i=1}^n$ with $V_i < V_{i+1}, \forall i$; probabilities p_i (or frequencies f_i) of occurrence of V_i; and two indices i, j of the pair of values (V_i, V_j) to be considered for computing their similarity. The output is the value of P_{ij} for (V_i, V_j).

```
1.  u = 0.0; s = 0.0; d = |V_j − V_i|; t = 1; P = 0.0; P̄ = 0.0;
2.  if i = j then
3.          begin
4.              for k = 1 to n do
5.                  if p_k ≤ p_i then P ← P + p_k
6.          end
7.  else
8.           begin
9.              if i > j then
10.                 begin
11.                     k ← i ; i ← j ; j ← k
12.                 end
13.             q = 0.0; for k = i to j do q ← q + p_k
14.             for k = 1 to n do
15.             if V_k + d ≤ V_n and s < 1 then
16.                 begin
17.                     if k > 1 then u ← u + p_{k−1}
18.                     while t ≤ n and |V_t − V_k| ≤ d do
19.                         begin
20.                             s ← s + p_t; t ← t + 1
21.                         end
22.                     P̄ ← P̄ + 2p_k(1 − s)
23.                     if |V_{t−1} − V_k| = d and s − u > q then
24.                         P̄ ← P̄ + 2p_k p_{t−1}
25.                 end
26.             else break
27.             P ← 1 − P̄
28.         end
29. return P;
```

Fig. 1. An $O(n)$ time and $O(n)$ space complexity algorithm for computing P_{ij}

Table 1. Comparisons of efficiency by experiments on large datasets: Census Bureau

	Datasets						
Specification	SS1	SS2	SS3	SS4	SS5	SS6	Full Set
# instances (m)	500	1,000	1,500	2,000	5,000	10,000	199,523
Dataset size (MB)	0.2	0.5	0.9	1.1	2.6	5.2	103
# diff. num. values (n)	497	992	1,486	1,973	4,858	9,651	97,799
Time of LIBISWAS	67.3s	26m6.2s	1h46m31s	6h59m45s	>60h	n/a	n/a
Time of OURS	**0.1s**	**0.2s**	**0.3s**	**0.5s**	**2.8s**	**9.2s**	**36m26s**
Memory LIBISWAS (MB)	5.3	20.0	44.0	77.0	455.0	n/a	n/a
Memory OURS (MB)	**0.5**	**0.7**	**0.9**	**1.1**	**2.1**	**3.4**	**64.0**
Preprocessing time	0.1s	0.1s	0.2s	0.5s	0.9s	6.2s	127.2s

Note that the body of the **while**-loop is executed in the total of n times on whole n steps of its outer **for**-loop, hence, the average number of executions of the **while**-loop's body on each step k is of $O(1)$, therefore, any **for**-loop of the algorithm is of $O(n)$. There are no nested **for**-loops, thus the algorithm is with the $O(n)$ time. The space is also of $O(n)$ because there are only 1-dimension arrays with indices from 1 to n in the algorithm.

4 Experimental Evaluation

Our experimental evaluation aims to two objectives: (i) to show that the proposed MSM can work on large datasets; and (ii) to investigate for which kinds of data the MSM can be used in distance-based methods to achieve good results. The proposed algorithm has been implemented in C and evaluated with large mixed datasets. We have also implemented Li and Biswas's algorithm to do comparisons of efficiency of the algorithms. All experiments were carried out on a workstation Alpha 21264 (OS: Digital UNIX V4.0E; Clock frequency: 500 MHz; RAM: 2 GB). Two datasets have been used to evaluate new MSM efficiency:

1. The U.S. Census Bureau (<http://www.census.gov/main/www/access.html>) of 199,523 instances with 33 symbolic attributes and 8 numeric attributes, and the size of the dataset is of 103MB. It is found that the maximal number of unique numeric values for the numeric attribute *"instance weight"* nearly equals the number of instances (i.e., $n \approx m$).
2. KDD Cup 1999 (<http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>) with 485,790 instances (73MB) as 10% of full dataset described by 6 symbolic attributes and 35 numeric attributes.

We carried out an experimental comparative evaluation on efficiency of the proposed method (called the OURS) and the previous one of Li and Biswas (called the LIBISWAS) in [7]. We randomly took from each of these two datasets six subsets SS1; SS2; ...; SS6 of 500; 1,000; 1,500; 2,000; 5,000; and 10,000 instances, respectively. The experiment was designed as follows: from each of these subsets and the full dataset, find the nearest neighbor for an instance randomly taken from outside of the subset; do compare the memory size used and execution time needed for the OURS and the LIBISWAS to do this task. That is, for each running on the dataset of m instances, it was necessary to compute MSM between the new instance and each of m instances of the subset. Experiment results on these two datasets are shown in Table 1 and Table 2. These results make clearly the advantage of our proposed method. Note that for SS6 with $m = 10,000$ and the full dataset Census Bureau with $m = 199,523$ and $m = 485,790$ for 10% of KDD-Cup 99, the LIBISWAS cannot fit the program with the data into the RAM of 2 GB (with "n/a": not applicable), but the OURS needs only 64 MB and 276 MB due to its requirement of $O(n)$ of memory size. Therefore, the OURS could be used for very large databases. Note that the OURS works under an assumption that $V_1, V_2, \cdots, V_n$ have been sorted (i.e., $V_1 < V < V_2 < \cdots < V_n$). For large databases and for the first time reading, it is necessary to collect the

Table 2. Comparisons of efficiency by experiments on large datasets: KDD-cup99

Specification	Datasets						
	SS1	SS2	SS3	SS4	SS5	SS6	10% Full Set
# instances (m)	500	1,000	1,500	2,000	5,000	10,000	485,790
Data size (MB)	0.1	0.1	0.2	0.3	0.7	1.5	73.6
# diff. num. values (n)	276	557	867	1,147	2,238	2,889	36,185
Time of LIBISWAS	6.2s	1m28.4s	12m54.2s	1h23m32s	>10h	n/a	n/a
Time of OURS	**0.1s**	**0.2s**	**0.3s**	**0.4s**	**2.0s**	**4.8s**	**11m21.3s**
Memory LIBISWAS (MB)	5.1	10.0	19.0	30.0	225.0	486.0	n/a
Memory OURS (MB)	**0.5**	**1.1**	**1.7**	**1.9**	**4.8**	**6.9**	**276.0**
Preprocessing time	0.1s	0.1s	0.1s	0.3s	0.5s	4.3s	20.1s

values of each attribute, to count their frequencies, to encode and sort the values, and so on. The slowest in this processing is sorting, which is with the $O(n \log n)$ time. When $n \approx m$ the number of t satisfying Eq.(10) is of $O(n)$, hence, the total time from reading a new database until the final results of the NNR program with the OURS is of $O(m + mn + n \log n) = O(mn) = O(n^2)$, the LIBIS is with $O(m + n^3) = O(n^3)$. To evaluate the usefulness of MSM, the decision tree induction program C4.5 with discretization of numeric data [8]) and k-nearest neighbor method with this MSM have been compared on 18 datasets containing mixed symbolic and numeric data from the UCI repository of databases. These datasets are divided into two groups: one with datasets having more numeric attributes than symbolic ones, and vice-versa. The preliminary results reported in [3] suggest some hypotheses about meta-knowledge of the use of this MSM and discretization for a given task of classification:

1. When a database contains more numeric attributes than symbolic ones, it is better to use MSM than discretization. The predictive accuracy of k-NNR with this MSM is often higher than that of C4.5 using discretization. However, k-NNR requires much more time than C4.5 to calculate MSM in these cases;
2. When a database contains more symbolic attributes than numeric ones, it maybe better to do discretization and use classifiers with symbolic data, vice-versa. The predictive accuracy of of k-NNR with this MSM is often lower than that of C4.5 using discretization. However, the computational time for this MSM is much lower than that of cases of many numeric attributes.

5 Conclusions

We studied a mixed similarity measure (MSM), initially proposed by Goodall [1], and the improved computation by Li and Biswas [7]. This MSM has been shown to work well with mixed datasets, but its computational costs are $O(n^2 \log n^2)$ time complexity and $O(n^2)$ space complexity, therefore, it was not suitable for the large databases with large number n of unique values for a numeric feature/attribute. We developed a mathematical basis for computing the same

MSM, but with a new and fast algorithm of only $O(n)$ time and $O(n)$ space complexities ($O(n \log n)$ time for the fisrt use of each dataset). The new method makes MSM, used to be impractical for large datasets, applicable to very large databases. The experimental results on some large datasets have demonstrated the efficiency of the proposed computing method and the algorithm, and suggested some meta-knowledge on the use of MSM and discretization. Further investigating the MSM with the proposed computation method by applying it to KDD distance-based methods such as NNR, clustering, classification, and so on, for large databases is very promising future work.

References

1. Goodall, D.W.: A New Similarity Index Based On Probability. *Biometrics*, Vol. 22 (1966) 882–907.
2. Gowda, K.C., Diday, E.: Symbolic Clustering Using a New Similarity Measure. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 22, No. 2 (1992) 368–378.
3. Ho, T.B., Nguyen, N.B., Morita, T.: Study of a Mixed Similarity Measure for Classification and Clustering. *3th Pacific-Asia Conf. on Knowledge Discovery and Data Mining PAKDD'99*. Lecture Notes in Artificial Intelligence 1574. Springer-Verlag (1999) 375-379.
4. Huang, Z.: Clustering Large Data Sets With Mixed Numeric and Categorical Values. *KDD: Techniques and Application*. World Scientific (1997) 21–34.
5. Ichino, M., Yaguchi, H.: Generalized Minkowski Metrics for Mixed Feature-Type Data Analysis. *IEEE Trans. Systems, Man and Cybernetics*, Vol. 24 (1994) 679–709.
6. Lancaster, H.O.: The Combining of Probabilities Arising from Data in Discrete Distributions. *Biometrika*. Vol. 36 (1949) 370-382.
7. Li, C., Biswas, G.: Unsupervised Clustering with Mixed Numeric and Nominal Data - A New Similarity Based Agglomerative System. *KDD: Techniques and Application*. World Scientific (1997) 33–48.
8. Quinlan, J.R. *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.