IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 2, 2019, accepted December 27, 2019, date of publication January 20, 2020, date of current version February 4, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2968222*

# A Heuristic Approach for Finding Similarity Indexes of Multivariate Data Sets

**RAHIM KHAN[1,2], MUHAMMAD ZAKARYA[1], AYAZ ALI KHAN[1], IZAZ UR RAHMAN[1], MOHD AMIRUDDIN ABD RAHMAN[3], MUHAMMAD KHALIS ABDUL KARIM[3], AND MOHD SHAFIE MUSTAFA[3]**

[1]Department of Computer Science, Abdul Wali Khan University, Mardan 23200, Pakistan
[2]Faculty of Computer Sciences and Engineering, GIK Institute of Engineering Sciences and Technology, Swabi 23640, Pakistan
[3]Faculty of Science, Universiti Putra Malaysia, Serdang 43400, Malaysia

Corresponding author: Mohd Amiruddin Abd Rahman (mohdamir@upm.edu.my)

This work was supported in part by the Higher Education Commission (HEC), Pakistan, in part by the Abdul Wali Khan University Mardan (AWKUM), Pakistan, in part by the Faculty of Computer Science and Engineering, Ghulam Ishaq Khan Institute of Engineering Sciences and Technology, Swabi, Pakistan, and in part by the Ministry of Higher Education, Malaysia, under Fundamental Research Grant Scheme (FRGS) under Grant FRGS/1/2017/TK04/UPM/02/5.

**ABSTRACT** Multivariate data sets (MDSs), with enormous size and certain ratio of noise/outliers, are generated routinely in various application domains. A major issue, tightly coupled with these MDSs, is how to compute their similarity indexes with available resources in presence of noise/outliers - which is addressed with the development of both classical and non-metric based approaches. However, classical techniques are sensitive to outliers and most of the non-classical approaches are either problem/application specific or overlay complex. Therefore, the development of an efficient and reliable algorithm for MDSs, with minimum time and space complexity, is highly encouraged by the research community. In this paper, a non-metric based similarity measure algorithm, for MDSs, is presented that solves the aforementioned issues, particularly, noise and computational time, successfully. This technique finds the similarity indexes of noisy MDSs, of both equal and variable sizes, through utilizing minimum possible resources i.e., space and time. Experiments were conducted with both benchmark and real time MDSs for evaluating the proposed algorithm's performance against its rival algorithms, which are traditional dynamic programming based and sequential similarity measure algorithms. Experimental results show that the proposed scheme performs exceptionally well, in terms of time and space, than its counterpart algorithms and effectively tolerates a considerable portion of noisy data.

**INDEX TERMS** Similarity index, multivariate data set, outliers, the longest common subsequence.

## I. INTRODUCTION

Recent technological advancements, particularly in sensors and actuators, lead to the generation of enormous multivariate data sets (MDSs) in different application areas i.e., wireless sensor networks, internet of things (IoT), scientific experiments, industrial control processes, educational purpose testbeds, web and databases [1]. An MDS is defined as a set of related numbers or values associated with a specific entity in an organization. In other words, a group of univariate data sets in columns form is known as MDS [2]. Mathematically, it is represented as a matrix $X_{m,n}$, where m and n corresponds to the rows and columns respectively. These MDSs are thoroughly examined, using various classical and non-classical approaches, to discover valuable information that is used to determine the correlating or distinguishing factor of entities. One of the major issue, closely linked with MDS, is to find their similarity indexes in the presence of noise/outliers that is not possible with existing techniques. Generally, two MDSs, $X_{i,j}$ and $Y_{m,n}$, are believed similar if most of their elements are highly correlated [3].

MDSs similarity problem is an active research area, both in computer science and mathematics, that is due to its existence in different real world application environments i.e., DNA analysis, sensors-based real-time decision support systems (DSS), computational biology, pattern matching, file comparison and etc. The LCSS problem is described as follows.

The associate editor coordinating the review of this manuscript and approving it for publication was Chongsheng Zhang.

*Definition 1: Given two MDSs* $S = S_{i,1}, S_{i,2}, S_{i,3}, ....S_{i,n}$ *and* $T = T_{i,1}, T_{i,2}, T_{i,3}, \ldots, T_{i,m}$ *where* $i$ *represents the dimensionality, the LCSS problem is to find a substring or subsequence of maximum length L such that L represents the identical substrings in both S and T i.e.,* $S_{i,n_1}$ *to* $S_{i,n_L}$ *and* $T_{i,m_1}$ *to* $T_{i,m_L}$ *where* $S_{i,n_1,\ldots,i,n_L}$ *and* $T_{i,m_1,\ldots,i,m_L}$ *are increasing order. Moreover, any two strings in the same sequence do not overlap with each other.*

*Definition 2: Given two MDSs* $S = S_{i,1}, S_{i,2}, S_{i,3}, ....S_{i,n}$ *and* $T = T_{i,1}, T_{i,2}, T_{i,3}, ....T_{i,m}$ *where* $i$ *represents the dimensionality, any two elements of S and T, that is* $S_{i,1}, S_{i,2}, S_{i,3}, \ldots, S_{i,n}$ *and* $T_{i,1}, T_{i,2}, T_{i,3}, \ldots, T_{i,m}$ *are considered similar* ***iff*** $S_{1,1}, S_{1,2}, S_{1,3}, \ldots, S_{1,n} = T_{1,1}, T_{1,2}, T_{1,3}, \ldots, T_{1,m}$ *where* $m = n$.

*Definition 3: Two MDs* $S_{i,n}$ *and* $T_{i,m}$ *are considered data sets with high similarity indexes* ***iff*** *their longest common subsequence length is such that* $LCSS_{length} > S_{3\times length/4}$ *or* $T_{3\times length/4}$.

In the literature, different techniques were presented, specifically for univariate data sets and MDSs, to solve this issue. Preliminary studies were focused on the idea of how to use the existing techniques, known as classical approaches, in solving the aforementioned problem. Euclidean distance measure was used to find the similarity index of univariate data sets but its results was trustworthy, particularly, in noise free data sets [4], [5]. Due to its noise sensitivity, euclidean distance measure is not a realistic solution for finding similarity index of both MDSs and univariate data sets. Similarly, due to the noise sensitive nature of the classical approaches, their implementation in realistic environments of MDSs is risky. Therefore, non-classical approaches (non-metric based approaches) were addressed, in literature, to resolve similarity issue of noisy MDSs and univariate data sets [6].

Non-metric based approaches, also known as longest common subsequence (LCSS), are ideal solutions for MDSs in general, and particularly for noisy MDSs. It is due to their non-sensitive nature to outliers or noise associated with MDSs that is usually avoided, in LCSS, by finding similarity index of the closest values only. In this regard, dynamic programming based similarity measure was an exceptional approach used to resolve this issue with available resources. However, this approach was developed for uni-variate data sets and is not suitable or ideal for finding similarity of MDSs [7]. To address this issue, various extensions of the tradition dynamic programming based algorithm (DPA) were presented to make it a realistic solution for both univariate data sets and MDSs, but most of these techniques were either data specific or overlay complex. Moreover, DPA based LCSS schemes were addressed to resolve the k-length substring problems [8]. However, most of these algorithms are designed specifically for uni-variate data sets and it is a tedious and time-consuming task to extend their operational capabilities to the MDSs. Moreover, a non-metric base approach for MDSs was presented in [9]. This approach divides the MDS into uni-variate sequences and starts computing their similarity in a sequential manner i.e., MDS S and T are divided in to sequence $S_1, S_2, S_3, \ldots, S_i$ and $T_1, T_2, T_3, \ldots, T_j$ respectively. Initially, LCSS of $S_1$ and $T_1$ is derived and stored with the location information of the matching elements in MDS. To reduce the time and space complexity, an element of sequence $S_2$ is matched with that of $T_2$ **iff** it resides in one of the locations that are stored in the previous step. This process is repeated for the remaining sequences of both S and T. However, the computational cost of this approach is higher than other field proven approaches particularly in situations where dissimilar elements reside in the last sequences or dimensions of the MDSs.

An alternative approach to the non-metric based solutions was the dynamic time warping (DTW) that is used to stretch the MDSs with respect to time [10]. DTW is used to find an optimal match between two sequences $S_i$ and $T_j$ subjected to some restrictions and rules such as $1^{st}$ element of S must be matched with $1^{st}$ element of T and the mapping of indices must be monotonically increasing, where i and j represents lengths of S and T respectively. Additionally, principal component analysis (PCA) based techniques were addressed to resolve the MDSs similarity problem by applying various procedures to extract valuable components and discard supplementary information [11]–[13]. Although, PCA and DTW work fine in different realistic environments of MDSs, but DTW and PCA have compromised on the noise sensitivity and information loss respectively.

In this paper, a computationally efficient multivariate similarity measure algorithm is presented to address the aforementioned issues specifically outliers, information loss, time and space complexities. The proposed scheme finds the longest common subsequence (LCSS) of two MDSs as follows.

1) Heuristic methodology is used to find similarity indexes of MDSs.
2) $\delta$ is used to limit the matching region or space of an individual element $S_{i,n}$ of an MDS in another MDS.
3) Similarity indexes of any two MDSs is subjugated to the **Definition 1**
4) Similarity of an individual element in an MDS is subjected to the **Definition 2**.

The rest of the paper is organized as follows. In section II, a brief overview of the literature is presented. In section IV, the pre-processing procedure is described. The proposed algorithm and its working mechanism is presented in section V followed by mathematical background. Section VI describes the results of the proposed algorithm and several field proven algorithms using benchmark data sets. Finally, concluding remarks and future research recommendations are given in section VII.

## II. LITERATURE REVIEW

Due to the technological advances, MDSs are generated in different application areas such as smart buildings, smart cities, wireless sensor networks, Internet of things (IoTs), scientific experiments, ECG signals and DNA analysis, stock

markets, multimedia, and industrial domains etc., [14], [15]. A tightly coupled issue with these data sets is how to determine their similarity indexes with a minimum possible computational time of the resources [16]–[18]. In the literature, different methods were proposed to solve the longest common subsequence problem particularly for multivariate data sets [19]. Some of these techniques are described below.

Euclidean distance measure was used by Chiu *et al.* [20] to retrieve a one dimensional time series motifs, that is defined as a region or subsequence of protein or DNA sequence that has a specific structure, of the underlined MDSs. Similarly, Mueen *et al.* [21] proposed a tractable exact algorithm to discover these motifs with a three order of magnitude faster speed than other existing approaches. Moreover, a series of techniques were proposed in literature to speed up the motifs discovery process [22]–[24]. A weighted function based similarity measure algorithm is presented by Xun *et al.* [25]. Initially, a function $f(x)$ is defined on the basis of a matrix external structure such as number of samples and range of the MDS. Then another function $g(x)$ is defined on the internal factors of the MDSs such as column vector correlation and weighted square norm. However, outliers sensitivity and complexity, particularly with maximum dimensionality and size of the MDSs, are among the closely linked issues with these techniques.

Non-metric based similarity approach is an alternative solution to find the similarity indexes of MDSs specifically in presence of outliers. A dynamic programming based LCSS computation algorithm, specifically for the k-length substring problems, was presented in literature to address the aforementioned issue i.e., outliers sensitivity [26], [27]. Zhu *et al.* [28] presented two different approaches to solve the LCSS problem with minimum possible time and space complexities **iff** $n = m$, where n and m represent sequence length. These algorithms perform well on uni-variate data sets, but their efficiency is degraded drastically if the underlying data sets are MDSs. Similarly, Yohei et al. used the LCSS concept to find similarity indexes in k-length order isomorphic substrings [29]. These techniques performed exceptionally well on smaller data sets, but their performance is severely affected (negatively) by a slight increase either in value of k or MDSs or both. A sequential LCCS computation mechanism for multivariate data sets similarity indexes was presented in [9] and claimed its exceptional performance on both real time and benchmark MDSs in the presence of outliers. This technique finds the LCSS of MDSs by dividing them into a series of uni-variate data sets. Yet, this algorithm suffers greatly in terms unnecessary computational time, particularly in MDSs where the ratio of dissimilar indexes of symbol is negligible at the starting dimensions, i.e., $row_1, row_2, row_3, \ldots, row_k$, and quiet high at the last i.e., $row_{n-k+1}, row_{n-k+2}, row_n$ where $k < n$.

In addition to the non-metric approaches, principal component analysis (PCA) was introduced to reduce dimensionality of the MDS by omitting their unnecessary or useless dimensions and concentrating on the adequate and mandatory principal components (PCs) [30]. Bootstrap re-sampling is an alternative approach which is used to determine mandatory components or PCs of a PCA method [31]. PCA is applied in a systematic way to preserve the overall integrity and authenticity of the original MDS within minimum possible PCs. Keogh *et al.* [32] addressed the potentials of PCAs particularly as a feature selection tool to compute the similarity indexes of MDSs. Generally, PCA is used to find the appropriate column or PCs and then another distance measure, i.e., Euclidean distance, is utilized to compute the desired similarity. Yang *et al.* [33] proposed a novel similarity measure, that was a hybrid of extended frobenius norm eros and PCA, to find the similarity indexes of MDSs. Although, this method works exceptionally well on different MDSs but its complexity analysis is the major issue. Outliers handling is one the closely linked issues associated with PCA based similarity methods. To resolve this, a robust PCA technique, that is a hybrid of $l_1$ norm and nuclear norm, was proposed to enable the recovery of a PCA method specifically situations where fraction of MDSs is corrupt or missing [34]. However, over and under estimation of an appropriate number of PCs for a specific MDSs and information loss are still the challenging task for the research community.

Vlachos *et al.* [35] presented an algorithm to find the trajectories of objects, specifically up to three-dimensional space, from MDSs. These trajectories are classified automatically by using the K-Nearest Neighbour (KNN) classification technique and then their similarity indexes is find by utilizing a non metric based approach, LCSS. This algorithm performs considerably well on uni-variate data sets, but its complexity is directly proportional to the dimensionality of the data sets. Similarly, handling of parallel movements is difficulty in higher dimensions. Likewise, A novel spatial index, i.e., GridVoronoi, is used to find the spatial nearest neighbour of two-dimensional datasets [36]. A similarity measure algorithm, that is based on the idea of dynamic programming algorithm proposed in [37], is presented to find similarity of the MDSs [38]. However, its efficiency and applicability is subjected to the heterogeneous parameters of MDSs and their smaller time constrains.

A well-known technique, i.e., dynamic time warping (DTW), is utilized to find the similarity indexes of MDSs in [39]. DTW stretches the MDSs along time-axis and restricts warping paths i.e., each point should be ordered monotonically, warping windows, continuous, slope constraints and boundary conditions. Initially, DTW was used in speech recognition area to mitigate the shortcomings which are introduced by the inflexibility along time-axis and then, used to find similarity of MDSs. Likewise, DTW based algorithm was presented in [40] to compute similarity of sequences, but its performance and efficiency on MDSs is not explored yet. These techniques work fine, but complexity and sensitivity to outliers are closely linked issued associated with DTW based techniques. Conclusively, various techniques have been presented in literature to measure similarity indexes of both uni-variate data sets and MDS, but similarity

of MDSs particularly with minimum possible time and space complexities is still an open research problem.

## III. PROBLEM STATEMENT

Similarity of MDSs, both sequence and numeric based data sets, is a challenging issue for the research community. Existing approaches, which are specifically designed for MDSs, do not take into account the core issue of unnecessary comparisons made to create similarity indexing table as in dynamic programming based algorithms i.e., matching $1^{st}$ element of MDS $S_{i,d}$ with last element of MDS $T_{j,d}$ and vice versa where $i = 1, 2, 3, ...n$, $j = 1, 2, 3, ...m$ and d represents their dimensions. Therefore, a non-metric based approach will resolve this issue if a sliding window based concept or time control parameter is adopted to shorten matching space or region of every symbol i.e., limiting the matching window of an element of MDS $S_{i,d}$ from i to $\delta$, a very small number, in another MDS $T_{j,d}$. For two MDSs $S_{i,d}$ and $T_{j,d}$, an element $S_{1,d} \in S_{i,d}$ is matched with elements of $T_{j,d}$ **iff** $T_{1,d} \in T_{j,d}$ and $T_{1,d} \in Window_{matched}$. Otherwise, next element of $S_{i,d}$ is loaded and processed accordingly.

## IV. PROPOSED METHODOLOGY: AN OVERVIEW

Generally in literature, solutions for the common LCSS problem is based on a well-known concept, i.e., dynamic programming, which matches every symbol of a sequence, i.e., $a_i$, where $a_i \in S_n$, with symbols of T, i.e., $b_j$, where $b_j \in T_m$, to generate a similarity matrix of size $m \times n$. This matrix $M_{i,j}$ contains information about symbols of both sequences, i.e., $S_n$ and $T_m$, which are matched in an increasing order and then $M_{i,j}$ is used to find their LCSS. A common problem associated with existing dynamic programming based techniques is their time and space complexities as LCSS, specifically for MDSs, is an NP-hard problem [28]. In this paper, we have focused on the specific issue, time and space complexity of non-metric based approaches, and presented a computationally efficient algorithm to measure similarity indexes of MDSs particularly in the presence of outliers. Additionally, the proposed mechanism does not require equality, in terms of length, of the MDSs.

### A. NORMALIZATION OF MDSs

MDSs are sequences of observations which are generated simultaneously in various application areas. These MDSs have different range of values, specifically those belong to the same application area. For example in stock market, the stock prices for the month of January lie within the range of 30 – 40 million PKRs whereas in December these prices rise to the range of 60 – 70 million PKRs. Therefore, to find similarity indexes other measures these MDS must be normalized to streamline their operations. In the literature, various normalization techniques were addressed, however, we have utilized the following normalization process due to its simplicity.

$$T_i = \frac{T_i - T_{min}}{T_{max} - T_{min}} \tag{1}$$

where, $T_i$ represents time series and $T_{max}$, $T_{min}$ represent maximum and minimum values of $T_i$, respectively.

### B. GENERAL PROCESS OF THE PROPOSED MECHANISM

Initially for given two MDSs as described in **Definition 1**, the proposed approach defines the sliding window control parameter $\delta$ that is used to bound the matching space or window of a particular element of an MDS, i.e., $a_i \in S_n$, within another MDS, i.e., $b_1, b_2, \ldots, b_k \in T_m$, where $k < m$ and $n \leq m$. Then in phase-I, the first symbol of S, i.e., $a_1 \in S_n$, is matched with symbols of T, i.e., $b_1, b_2, \ldots, b_k \in T_m$, such that a match is encountered or limit of the underlying sliding window is reached i.e., $b_k$ is not matched. If a symbol of S, *i.e.*, $a_i \in S_n$, matches an element of T,*i.e.*, $b_j \in T_m$, then either $a_i$ or $b_j$ is stored in an array $LCSS_{matched}$ with their location information i.e., index or position in T. However, if a symbol $a_i \in S_n$ does not match with any element of T, i.e., $b_1, b_2, \ldots, b_k \in T_m$, within a defined window, then $a_i$ is neglected and next symbol from S $a_{i+1} \in S_n$ is selected. Next element of S, i.e., $a_2 \in S$, is loaded and matched with every symbol of T such that the matching process starts exactly after previously matched location in T i.e., $b_p$, where p represent previous matched position. Position information helps in the implementation of various restrictions on the computed LCSS i.e., increasing order as described in **Definition 1** of section I. This mechanism is repeatedly applied to the remaining symbols of S, $a_2, a_3, \ldots, a_n \in S_n$, such that symbol is added to $LCSS_{matched}$ **iff** $a_i \in S_n == b_j \in T_m$ and discarded otherwise. In phase-II, MDS S is revised such that the first symbol of S, i.e., $a_1 \in S_n$, is discarded **iff** it is part of the $LCSS_{matched}$ and the remaining portion of S, i.e., $a_2, a_3, \ldots, a_n \in S_n$, is considered refined MDS. The aforementioned process, i.e., computation of $LCSS_{matched}$, is repeated for the revised MDS i.e., S. The length of current $LCSS_{matched}$ is matched with the previous $LCSS_{matched}$, the one $LCSS_{matched}$ with maximum length is retained while other is discarded. This process is repeated until the desired LCSS is found, i.e., in cases where length of the desired LCSS is known, or when 1/3 or 1/2 of the S is encountered.

To understand the proposed idea, let us assume two sequences $S_n$ and $T_m$ such that $S_n$ = AEBACFDADB and $T_m$ = CABDACDADB where n = m = 10 and $\delta = 3$. First symbol $a_1 \in S_n$ of $S$, that is A in this case, is matched with every symbol of $T$, i.e., $b_1, b_2..b_3 \in T$ which is CAB, within its bounded window, i.e., $\delta = 3$ and adds symbol A to the $LCSS_{matched}$ class with its location information in MDS T i.e., 2 in this case. Then next symbol of S $a_2 \in S$, i.e., E, is loaded and matched with every symbol of T just after the previous matched location, i.e., $b_2 \in T$ that is A at position-2 in T, within specified range of the $\delta$. Since, E does not match any symbol of T with specified matching window, hence, it is discarded and next element is selected from S i.e., $a_3 \in S$ which is B. B is matched with elements of T after the previous matched location i.e., 2. A match is encountered at position-3 in MDS T. B is stored in $LCSS_{matched}$ with its location information that is 3. Likewise, this process is repeated for

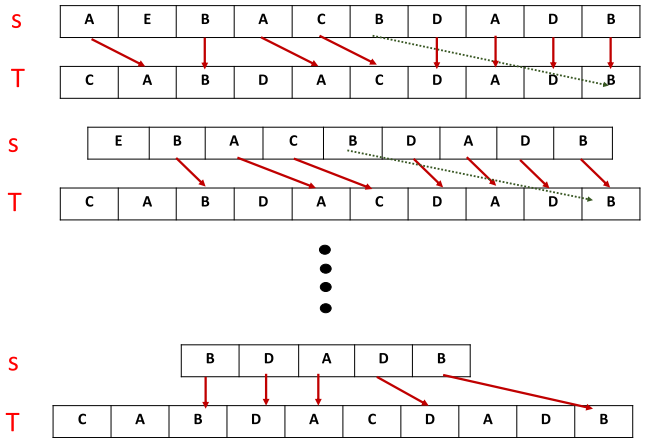


**FIGURE 1.** Graphical Representation of the Proposed LCSS Computation Mechanism for MDSs.

the remaining symbols of S until their LCSS is computed i.e., ABACDADB. Phase-II is applicable in those scenarios or MDSs where length of the computed LCSS, $LCSS_{matched}$, is less than length of shorter MDS/2 i.e., length($LCSS_{matched}$) $<$ length(Shorter-MDS/2). A graphical representation of the proposed LCSS computation mechanism is presented in Figure. 1. For simplicity and understandability, the graphical representation is limited to the matching process of two sequence S and T.

## V. PROPOSED METHODOLOGY

Non-metric based approaches, i.e., LCSS, are widely used to find similarity indexes of both uni-variate and MDSs specifically in the presence of outliers and noise. In literature, traditional dynamic programming based algorithm is a legitimate way to describe the similarity indexes of the underlying data sets. Although, this approach works fine on various data sets, but it does not consider the properties of MDSs i.e., if similarity index is 100% or 0%, same resources are used. Assume that we have two MDSs S and T such that S $= a_1, a_2, a_3, ...a_n$ and T $= b_1, b_2, b_3, \ldots, b_m$ where $a_1 == b_1$, $a_2 == b_2$,..., $a_n == b_m$, existing dynamic programming based algorithms generate a similarity matrix $M_{i,j}$ where matching of symbol $a_1$ of S with symbol $b_m$ of T, $a_n$ with $b_1$ and so on are performed. Although, these matches seem useless or wastage of resources to a naive user, but these are mandatory steps to develop an appropriate matrix $M_{i,j}$. Time and space complexities are the main issues which are tightly coupled with existing non-metric based approaches. To resolve these issue, a computationally efficient algorithm is presented to compute LCSS of the MDS by mitigating the unnecessary comparisons and hence their similarity.

### A. PROPOSED HEURISTIC APPROACH FOR FINDING LCSS WITH EXAMPLE

Consider MDs $S_{10}$ and $T_{10}$ given in Table. 1 and Table. 2. The proposed algorithm finds their longest common subsequence $LCSS_{matched}$ by matching the

**TABLE 1.** Multivariate data set S.

| Date and Time | Temp $C^0$ | Humidity | Moisture |
|---|---|---|---|
| 15/4/2011 22:00 | 35 | 92 | 780 |
| 15/4/2011 22:15 | 39 | 82 | 778 |
| 15/4/2011 22:30 | 36 | 87 | 776 |
| 15/4/2011 22:45 | 35 | 91 | 774 |
| 15/4/2011 23:00 | 37 | 85 | 772 |
| 15/4/2011 23:15 | 36 | 87 | 772 |
| 15/4/2011 23:30 | 38 | 83 | 770 |
| 15/4/2011 23:45 | 35 | 90 | 767 |
| 15/4/2011 00:00 | 38 | 82 | 762 |
| 15/4/2011 00:15 | 36 | 86 | 756 |

**TABLE 2.** Multivariate data set T.

| Date and Time | Temp $C^0$ | Humidity | Moisture |
|---|---|---|---|
| 16/4/2011 2:00 | 37 | 85 | 782 |
| 16/4/2011 2:15 | 35 | 92 | 780 |
| 16/4/2011 2:30 | 36 | 87 | 776 |
| 16/4/2011 2:45 | 38 | 84 | 775 |
| 16/4/2011 3:00 | 35 | 91 | 774 |
| 16/4/2011 3:15 | 37 | 85 | 772 |
| 16/4/2011 3:30 | 38 | 83 | 770 |
| 16/4/2011 3:45 | 35 | 90 | 767 |
| 17/4/2011 0:00 | 38 | 81 | 762 |
| 17/4/2011 0:15 | 36 | 86 | 756 |

corresponding elements of S with those of T i.e., $a_i$ (Temperature) is matched with $b_j$ (Temperature), $a_{i+1}$ (Humidity) with $b_{j+1}$ (Humidity),...$a_{i+n}$ (Moisture) with $b_{j+n}$ (Moisture), where i = j = 1. In phase-I, $row_1$ of S, i.e., $S_{1,temperature}, S_{2,humidity}$ and $S_{3,moisture}$, is matched with $row_{1\ tok}$ of T, i.e., $T_{1\cdots k,temperature}, T_{2\cdots k,humidity}$ and $T_{3\cdots k,moisture}$ where $k == \delta$, until a match is encountered or window control parameter, i.e., $\delta$, terminates this process. In scenarios where a match is encountered, a pointer $p_i$ to the matched $row_j$ of MDS T is stored in $LCSS_{matched}$ and the subsequent $row_{i+1}$ of S is selected. However, the matching process does not start from the $row_1$ of T, rather it is continued from the previously matched location i.e., $p_j$. Conversely, if a $row_i$ of S does match with any $rowj\cdots k$ of T, then next $row_{i+1}$ is selected for processing. This process is repeated for the remaining rows $row_i$ of S and ends with a computed LCSS that is stored in $LCCS_{req}$. In phase-II, $row_i$ of S is deleted temporarily **iff** it appears in the computed $LCSS_{req}$ as a first symbol to look for the alternative solutions i.e., LCSS. After deleting $row_i$, the matching process is initiated with the first row, $row_{i,new}$, of the updated S. The aforementioned process to find $LCSS_{matched}$ is repeated for the updated S and then the computed $LCSS_{matched}$ length is compared with that of previous $LCSS_{req}$. $LCSS_{req}$ holds LCSS with maximum length. Phase-II is repeated for every $row_{i\cdots M}$ of S which appeared in the $LCSS_{req}$ until the desired LCSS is computed or M is encountered, where M represents middle of S.

To understand this idea, consider MDSs S and T as described above and assume that the window control parameter value is set to three i.e., $\delta = 3$. $Row_1$ of S is matched with each $row_{j\cdots j+3}$ of T until a match is encountered, i.e., $row_1$ of S is matched with $row_2$ of T at time 15/4/2011 22:00 and 16/4/2011 2:15 of S and T respectively such that $S_{temperature(row_1)} = 35^0 = T_{temperature(row_2)}$, $S_{humidity(row_1)} = 92 = T_{humidity(row_2)}$ and $S_{moisture(row_1)} = 780 = T_{moisture(row_2)}$. A pointer to $row_2$ of T is added to the $LCSS_{matched}$ and next $row_2$ of S is loaded for further processing. It is to be noted that $row_2$ matching window does not start with $row_1$ of T, but it is continued from the previously matched position i.e., $row_3$. $Row_2$ does not have a match in the T, therefore, next $row_3$ of S is selected. It matches with $row_3$ of T and a pointer to $row_3$ of T is added to the $LCSS_{matched}$. This process is repeatedly applied to the remaining rows $row_{3\cdots n}$ of S until the desired $LCSS_{matched}$ is found or $row_n$ of S is encountered, then the contents of $LCSS_{matched}$ is copied to $LCSS_{req}$. In phase-II, since $row_1$ of S is part of the $LCSS_{matched}$, therefore, it is temporarily deleted from S to form a modified S such that $row_S > row_{S_{modified}}$. For the modified S, the aforementioned process is repeatedly applied to find another $LCSS_{matched}$ that is matched with previously stored $LCSS_{req}$. $LCSS_{req}$ is updated **iff** length($LCSS_{matched}$) > length($LCSS_{req}$). This process is repeated until required $LCSS_{req}$ is computed or M is encountered.

### B. PROPOSED LCSS ALGORITHM SPECIFICALLY FOR MDSS

Variable M represents the $3/4^{th}$ value or index of the MDS i.e., S in this case whereas TP is used to store the position of matched elements in time series T, as it is necessary to avoid unnecessary comparisons i.e., successor symbol of an already matched $S_i$ with the predecessor of $T_j$ such that $S_i = T_j$. Variables *Count* and $Count_1$ are used to store length information of the currently and previously computed LCSS respectively. $\delta$ is sliding window control parameter that is used to bound the matching window of a particular symbol and LCSS is an array to store the computed LCSS.

**Algorithm 1** Proposed LCSS Computational Algorithm for MDSs

**Input**: Longest Common Subsequence of MDSs
**Output**: Return the Longest Common Subsequence (*LCSS*)

```
LCSS(S_n, T_m);
Pre-LCSS ← 0;
Cur-LCSS ← 0;
TP ← 0;
Count ← 0;
Count_1 ← 0;
for a ← 0 to M do
    for i ← a to S_n do
        for j ← TP to T_m AND a_i ∈ δ do
            if S or T = 0 then
                LCSS ← 0;
            else
                if a_{i,n} == b_{j,m} then
                    Class_1 ← a_{i,n} or b_{j,m};
                    TP ← j + 1;
                    Count ← Count + 1;
                    break;
                else
                    if a_{i,n} ≠ b_{j,m} then
                        Next element;
                    end if
                end if
            end if
        end for
    end for
    if Count > Count_1 then
        Pre-LCSS ← Cur-LCSS;
        Count_1 ← Count;
        Count ← 0;
    end if
    Delete the a^th element from S_n;
    TP ← 0;
end for
return LCSS of MDSs S and T
```

### C. ANALYSIS OF THE PROPOSED SCHEME

Multivariate data sets (MDSs) $S_{n,i}$ is defined a set of related values such that $a_{1,1}, a_{1,2}, \ldots, a_{1,i}, a_{2,1}, a_{2,2}, \ldots, a_{2,i}, \ldots, a_{n,1}, a_{n,2}, \ldots, a_{n,i} \in S_{n,i}$ where n, i represent length and dimension of the data sets respectively. The longest common subsequence is represented by $LCSS(k, l)$ where $k \leq n$ and $l \leq i$. Concatenation appending of two symbols is of the form XY such that $X \in S$ and $Y \in S$.

Definitions of the basic terms are as described below.

*Definition 4: Any two symbols $X \in S$ and $Y \in T$ are equal* ***iff*** *distance(X,Y) = 0.*

*Definition 5: The $LCSS(0, 0)$ is the description of an empty LCSS.*

*Definition 6: A $row_i$ $a_{1,p}, a_{2,p}, a_{3,p}, \ldots, a_{n,p} \in S$ is considered predecessor of another $row_{i+t}$ $a_{1,q}, a_{2,q}, a_{3,q}, \ldots, a_{n,q} \in S$ in an $LCSS_{matched}$,* ***if*** *index($p > q$) and for both $row_i$ and $row_{i+t}$, $\exists$ ($row_j$ and $row_{j+r}$) $\in T_m$ such that $row_i == row_j$ and $row_{i+t} == row_{j+r}$ $\forall$ $i$ & $j = 1, 2, 3, \ldots, n$ where t and r represent any number.*

*Definition 7: A $row_i$ $a_{1,p}, a_{2,p}, a_{3,p}, \ldots, a_{n,p} \in S$ is not considered predecessor of another $row_{ii}$ $a_{1,q}, a_{2,q}, a_{3,q}, \ldots, a_{n,q} \in S \notin LCSS_{matched}$,* ***if*** *index($p < q$) that is $i > ii$ and, although, for both $row_i$ and $row_{ii}$, $\exists$ ($row_j$ and $row_{j+r}$) $\in T_m$ such that $row_i == row_j$ and $row_{ii} == row_{j+r}$ $\forall$ $i$ & $j = 1, 2, 3, \ldots, n$ where ii and r represent any number.*

*Definition 8: The $LCSS(k, l)$ is the LCSS of MDSs S and T if $a_{n,i} == b_{m,j}$ and there exist $i' < i$ and $j' < j$ such that $LCSS(k - i', l - j')$ is generated by $i'$ and $j'$.*

*Lemma 1: For $k >= 1$, $LCSS(p, l)$ is the longest common subsequence of MDSs S and T of length m and n respectively*

***iff*** *there exist some* $a_{1\cdots p,l} == b_{1\cdots p,l} \because a_{1\cdots p,l} \in S$ *and* $b_{1\cdots p,l} \in T$ *where* $p \leq n$ *&* $m$.

*Proof:* Applying mathematical induction on $k$. The $length(LCSS(1,l))$ is equal to 1 **iff** $a_{1,l} = b_{1,l}$ (According to **Definition 6**), that finds LCSS of length 1. Hence, the lemma is true for $k = 1$. Assume that the lemma holds for the value of $k-1$ and we need to prove that it holds true for the value of $k$. If $LCSS(i,j)$ is the LCSS of $S_{i,l}$ and $T_{j,l}$ then there exists $i' < i$ and $j' < j$ such that $LCSS(i',j')$ is the LCSS of $S_{i',l}$ and $T_{j',l}$ for the value of $k-1$. According to our assumption, $LCSS(i',j') = p'_{1,l}, p'_{2,l}, p'_{3,l}, \ldots, p'_{k,l}$ such that $p' < p$ and $p'_{1,l}, p'_{2,l}, p'_{3,l}, \ldots, p'_{k,l} \in S$ & $T \because a_i = b_j \because i < n$ and $j < m$. Therefore, LCSS of $S_{n,l}$ and $T_{m,l}$ is of length k $\because length(LCSS(i',j')) + length(LCSS(i,j)) = k$. Hence, it proves that length($LCSS(p,l)$) is $k$. Conversely, if $length(LCS(i,j)) \geq k$ and $a_i = b_j$, where $a_i \in S$ and $b_j \in T$ then there exist $i' < i$ and $j' < j$ such that $a_{i'} = b_{j'}$ and $length(LCSS(i',j')) = length(LCSS(i,j)) - 1 \geq k-1$. $LCSS(i',j')$ is the LCSS of $k-1$ length MDS (By inductive Hypothesis). Hence, $LCSS(p,l)$ is the desired LCSS. □

*Theorem 1: The LCSS problem can be solved in O(n) time; where n and m are the lengths of MDSs S and T respectively (the best case such that S ad T are identical either completely or nearly complete).*

*Proof:* According to the **Definition 2**, for any two MDSs such as S and T, any two elements of the underlying MDSs are considered similar **iff** $a_{1,1}, a_{1,2}, a_{1,3}, \ldots, a_{1,n} \in S == b_{1,1}, b_{1,2}, b_{1,3}, \ldots, b_{1,m} \in T$ where m,n represent MDSs lengths and $m = n$. Therefore, an element or symbol of an MD, i.e., $a_{1,1}, a_{1,2}, a_{1,3}, \ldots, a_{1,n} \in S$ that is a row, is added to the $LCSS_{matched}$ class only, if it fulfills the defined criteria i.e., if $b_1, b_2 \in LCSS_{matched}$, then $index(b_1) < index(b_2)$. Since, symbols of both MDSs are identical completely, therefore, every entry in the desired $LCSS_{matched}$ is performed after a fixed number of matches i.e., one or two (possibly in rare case). Outer loop of the proposed scheme, i.e., developed for MDSs $S_n$, is executed once whereas inner loop, i.e., developed for MDSs $T_m$, executes m times. Hence, $LCSS_{mathched}$ for the identical MDSs is computed in $O(m)$ time. □

*Theorem 2: The proposed MDSs algorithm solves the LCSS problem in O(k × n × δ) time; where n and m represent lengths of MDSs S and T respectively (the worst case such that S ad T are different either totally or partially).*

*Proof:* According to **Definition 3**, MDSs $S_n$ and $T_m$ are considered highly similar **iff** $LCSS_k(S_n, T_m) > S_{3\times n/4}$ or $T_{3\times n/4}$. Therefore, $LCSS_k(S_n, T_m) < S_{3\times n/4}$ or $T_{3\times n/4}$ is an indication of their dissimilarity metric. As a higher value of this metric is directly proportional to the computational complexity of the non-metric based approaches. ∴ for totally different MDSs, a single value of outer loop results in complete cycle, i.e., $\delta = suitablevalue$, of the inner loop that is ∀ $a \in S_n$, the loop for $T_m$ executes at-most $\delta$ times that is $O(\delta)$. The outer loop will be executed n times ∀ $(a_1, a_2, \ldots a_k) \in S_n$. ∴ ∀ MDSs $S_n$ and $T_m$ the proposed algorithm solves it in $O(M \times n \times \delta)$ such that $a_{1,1}, a_{1,2}, a_{1,3}, \ldots, a_{1,n} \in S \neq b_{1,1}, b_{1,2}, b_{1,3}, \ldots, b_{1,m} \in T$ or simply $\nexists$ any$(a_i \in S == b_j \in T)$, where M and $\delta$ are very small numbers. □

## VI. EXPERIMENTAL RESULTS AND EVALUATIONS

The proposed algorithm is tested on different evaluation metrics, i.e., complexity, running or execution time, comparisons and variable length data sets, against field proven schemes. Moreover, these algorithms are tested on both benchmark and real time MDS that is collected through our deployed WSN.

### A. COMPLEXITY ANALYSIS

Complexity analysis is necessary to elaborate the reliability, applicability, maintainability and robustness of an algorithm. In realistic scenario, complexity of an algorithm is directly proportional to its performance in target application areas particularly technological infrastructures. Hence, mostly, algorithms with minimum complexity are preferred over highly complex scheme, as it is evident from literature that the algorithms' selection criteria varies from application to application. However, it is observed that algorithms or schemes with minimum complexity is appreciated in different domains.

The best case, where both MDS such as S and T are either similar completely or having maximum similarity indexes, complexities of the proposed algorithm, sequential approach based, and dynamic programming based algorithms for MDS are $O(m)$, $O(m+h)$ and $O(m \times n + h)$ respectively where m and n represent MDS lengths and h is their dimensions or variables. The proposed scheme is an ideal solution particularly in this case, because it computes the similarity indexes, that is the longest common subsequence, of both MDS in one iteration where its rival schemes requires complete iterations of both loops i.e., to generate similarity metric, both MDS must be processed completely. Similarly, the worst case complexities of the proposed algorithm, sequential approached based and dynamic programming based algorithms are $O(M \times n \times \delta)$, $O(M \times n \times \delta) + O(h)$ and $O(m \times n \times h)$, respectively; where $\delta$ is an application dependent constant and M is a small number such as $\frac{1}{4}$, $\frac{1}{3}$ and $\frac{1}{2}$(S or T). Note that the time or window control parameter $\delta$ plays a vital role in improving the proposed scheme performance, specifically, in terms of comparisons, time and space needed or utilized.

The average case complexity of the proposed scheme is computed by dividing the problem into disjoint sets of sub problems and then solve these sub-problems by starting from the simplest one i.e., sub-problem with MDS of length two only. The procedure requires n-steps to resolve a particular sub-problem that is for $j = 0$ to $n$. Some of these sub-problems are solved by negligible work while other require maximum steps, i.e., n. Let's assume that the probability of a sub-problem $p_1$ which requires j steps for its completion is represented by $p_j$, then its average case complexity is given by equation 2:

$$T_{avg}(Algorithm) = \sum_{j=0}^{m} m \times p_j \tag{2}$$

where m represents the length of the subproblem. Let's assume that the probability of matched and not-matched symbols in MDS S and T is given by equations 3 and 4, respectively.

$$P_{mtd} = \delta \tag{3}$$
$$P_{nmtd} = \frac{1}{\delta} \tag{4}$$

Then, the total probability of a subproblem $P_1$ is, as described below.

$$P_1 = \frac{1}{\delta} + \frac{\delta - 1}{\delta} \tag{5}$$

Likewise, total probability of the subproblems $p_2, p_3, \ldots p_n$ of MDS S and T are given by the following equations.

$$P_2 = \frac{1}{2\delta} + \frac{2\delta - 1}{2\delta} \tag{6}$$
$$P_3 = \frac{1}{3\delta} + \frac{3\delta - 1}{3\delta} \tag{7}$$
$$\vdots$$
$$P_n = \frac{1}{n\delta} + \frac{n\delta - 1}{n\delta} \tag{8}$$

The average case complexity of the proposed algorithm is given by

$$T_{avg}(Pro - Algo) = \sum_{k=0}^{M} \sum_{i=0}^{m} \sum_{j=k}^{\delta} M \times n \times \delta \times P_k \times P_i \times P_j \tag{9}$$

putting values in equation 9, we get Eq. 10 and Eq. 11, as shown at the bottom of this page.

By applying algebraic manipulation, the average complexity of the proposed scheme is given by equation 12.

$$T_{avg}(Pro - Algo) = M \times n \times \delta \tag{12}$$

### B. COMPUTATIONAL TIME AND SPACE

The proposed algorithm along with its rival schemes are implemented in C++ development environment using similar resources, i.e., processor power & memory, and data sets, which are both real time and benchmark. A real time data set was obtained through the deployment of an operational wireless sensor network in real agriculture environment i.e., orange orchard [41]. Initially, these data sets were normalized using equation 1 that changes the values of numeric columns to a common scale i.e., usually between 0 and 1, without distorting differences in their ranges. Moreover, in case of datasets with sequences, such as matching A with A or B, an exact matching scenario is adopted i.e., $A = A$. However, for data sets with numeric values, similarity of any two values is subjected to a defined threshold value such as 0.01 or 0.02.

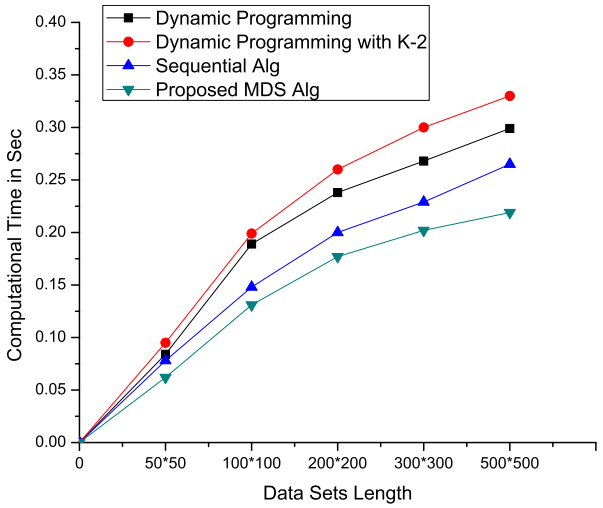


**FIGURE 2.** Computational Time of Proposed and Dynamic Programming based Algorithms on MDSs having Constant Length and Dynamic Variables.

### C. RESULTS AND DISCUSSION

It is clearly evident from Figure. 2 that the proposed algorithm requires the least possible computational time to compute LCSS of any two data sets, with a constant length and varying dimensionality, against field proven schemes i.e., sequential approach [9] and dynamic programming based algorithms [8], [27], [37]. Additionally, if similarity indexes of any two data sets, both real and benchmark, is high then performance of the proposed scheme is exceptionally well as shown in Figure. 3 such as if $S_i$ and $T_j$ are completely similar then the proposed approach computes their LCSS in $O(m)$ time, where m represents the data set with maximum length. An interesting feature of the proposed scheme is its non-sensitivity to variable/dynamic length and dimensionality of the data sets and its performance is, largely, consistent whereas dynamic programming based algorithms are highly sensitive. Apart from data sets with varying length and dimensionality, these schemes were also tested on fixed

$$T_{avg}(Pro - Algo) = M \times n \times \delta \times (\frac{1}{n} + \frac{1}{n}, \ldots, + \frac{1}{n}) \times (\frac{1}{n} + \frac{1}{n}, \ldots, + \frac{1}{n}) \times (\frac{1}{\delta} + \frac{\delta - 1}{\delta} + \frac{1}{2\delta} + \frac{2\delta - 1}{2\delta}, \ldots, + \frac{1}{n\delta} + \frac{n\delta - 1}{n\delta}) \tag{10}$$

$$T_{avg}(Pro - Algo) = M \times n \times \delta \times \frac{1}{n}(1 + 1, \ldots, +1) \times \frac{1}{n}(1 + 1, \ldots, +1) \times \frac{1}{\delta}(1 + \delta - 1 + \frac{1}{2} + \frac{2\delta - 1}{2}, \ldots, + \frac{1}{n} + \frac{n\delta - 1}{n}) \tag{11}$$

**TABLE 3.** Computational time (in seconds) of proposed, sequential and dynamic programming based algorithms on benchmark MDSs.

| Data Sets | DP-based Alg. | DP-based Alg. with K-2 | Sequential Alg. | Proposed MDS Alg. |
|---|---|---|---|---|
| Hobolink-500 | 1.1091 | 1.1880 | 0.9220 | 0.8750 |
| Amex-500 | 0.625 | 0.8060 | 0.5470 | 0.5310 |
| Robotic-300 | 0.2810 | 0.3170 | 0.2650 | 0.2380 |
| Haptitus-500 | 0.4220 | 0.4561 | 0.3910 | 0.3750 |
| Twopattern-500 | 1.1720 | 1.2138 | 0.7810 | 0.6870 |
| Bacteria-700 | 1.3910 | 1.7430 | 1.0470 | 0.9896 |

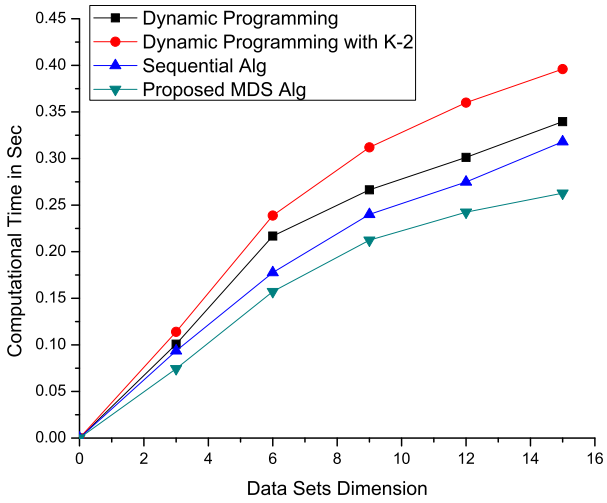


**FIGURE 3.** Computational Time of Proposed, sequential and Dynamic Programming based Algorithms on MDSs having Constant Variables and Dynamic Length.

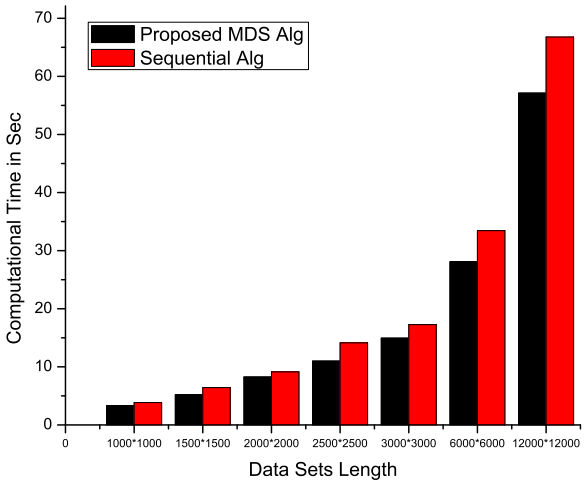


**FIGURE 4.** Computation Time of Proposed and Sequential Algorithms on larger MDSs.

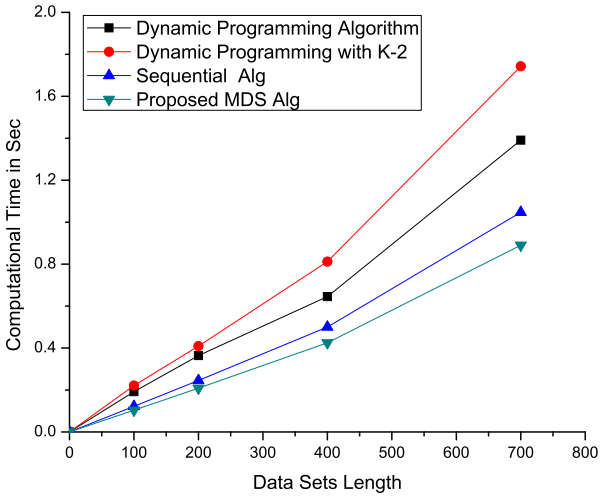


**FIGURE 5.** Computational Time of Proposed and Sequential Algorithms on MDSs of Constant Length and Dimensionality.

length and dimensionality as well as, usually, benchmark data sets are of constant or fixed length and dimensionality as shown in Figure. 5. Moreover, the proposed scheme performs better than the sequential approach on larger data sets with fixed length and dimensionality as shown in Figure. 4. Performance of the sequential approach is directly proportional to similarity indexes of the data sets and more importantly if those indexes reside in ending/last columns or dimensions whereas dynamic programming based algorithm performs unnecessary comparisons, i.e., matching first element $S_1$ of $S_i$ with last element of $T_j$, to create similarity indexes table. The proposed scheme resolves both of these issues such as it incorporates a time control parameter $\delta$ to avoid unnecessary comparisons. Likewise, a row-wise methodology is used to resolve the problem associated with sequential approach.

In order to verify the exceptional performance of our proposed scheme, these algorithms were tested on several publicly available benchmark data sets such as National Center for Biotechnology Information (NCBI) viral genomes [42], UCR time series repository [43], UCI (UCI KDD Archive, 2019) [44], American Stock Exchange and the Onset Computer Corporation live data feeds (HOBO U30 Remote Monitoring Systems 2015-16). The name of every data set is concatenated with its size such as Hobolink-500; where means that Hobolink data set with length of 500. Initially, these data sets were normalized and then used to verify our claims i.e., non-sensitivity and minimum computational time. It is evident from Table.3, that the proposed scheme has shown the consistency and superiority over sequential and dynamic programming based algorithms on different data sets. We have observed that the computational time of these algorithms is directly proportional to the similarity indexes in the underlined MDS as shown in Table. 3.

Although, the proposed scheme showed excellent performance on different MDSs but its efficiency and reliability is subjected to the proper selection of $\delta$.

## VII. CONCLUSION AND FUTURE WORK

Due to the non-sensitive nature to outliers, non-metric based approaches are most widely used methodologies to find similarity indexes of both uni-variate and multivariate data sets. However, existing approaches were either overly complex or require excessive computational and storage resources to accomplish this task i.e., finding similarity indexes of MDSs. To address this issue, a computationally efficient algorithm, that is developed for data sets in general and, particularly, for MDSs, is presented to depict how similar two MDSs are? A time or window control parameter $\delta$ is used to limit the matching region of an element or symbol in another data set. Therefore, unlike traditional dynamic programming based schemes, where first element of a data set is compared with the last element of another data set, the proposed scheme matches elements within a specific window. Matching in a limited region not only avoids unnecessary comparisons but also saves computational resources needed to accomplish this task, that is processor and memory. The proposed algorithm along with field proven algorithms were tested and evaluated on real data sets, both real-time and benchmark, of different sizes and dimensionality; because real-time MDSs are not always of the same size and dimensionality. Simulation results have verified the exceptional performance of the proposed scheme against its closest rival algorithms on several real and publicly available benchmark data sets.

In the future, we are eager in applying the proposed scheme to reduce noise, either at node or cluster head level, in an operational constraint oriented network such as wireless sensor networks or Internet of Things (IoTs). Moreover, we will extend our work to develop a reliable algorithm to compute gap-free LCSS specifically for both sequence oriented and numeric MDS.

## REFERENCES

[1] S. Chen, S. Zhang, X. Zheng, and X. Ruan, "Layered adaptive compression design for efficient data collection in industrial wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 129, pp. 37–45, Mar. 2019.
[2] K. ØMikalsen, F. M. Bianchi, C. Soguero-Ruiz, and R. Jenssen, "Time series cluster kernel for learning similarities between multivariate time series with missing data," *Pattern Recognit.*, vol. 76, pp. 569–581, Apr. 2018.
[3] C. Chatfield, *Introduction to Multivariate Analysis*. Evanston, IL, USA: Routledge, 2018.
[4] J. Draisma, E. Horobeţ, G. Ottaviani, B. Sturmfels, and R. R. Thomas, "The euclidean distance degree of an algebraic variety," *Found. Comput. Math.*, vol. 16, no. 1, pp. 99–149, Feb. 2016.
[5] P.-E. Danielsson, "Euclidean distance mapping," *Comput. Graph. Image Process.*, vol. 14, no. 3, pp. 227–248, Nov. 1980.
[6] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Mining Knowl. Discovery*, vol. 26, no. 2, pp. 275–309, Mar. 2013.
[7] Y. Li, H. Li, T. Duan, S. Wang, Z. Wang, and Y. Cheng, "A real linear and parallel multiple longest common subsequences (MLCS) algorithm," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2016, pp. 1725–1734 .
[8] G. Benson, A. Levy, and B. R. Shalom, "Longest common subsequence in $k$ length substrings," in *Proc. 6th Int. Conf. Similarity Search Appl. (SISAP)*, A Coruña, Spain, Oct. 2013, pp. 257–265.
[9] R. Khan, I. Ali, S. M. Altowaijri, M. Zakarya, A. Ur Rahman, I. Ahmedy, A. Khan, and A. Gani, "LCSS-based algorithm for computing multivariate data set similarity: A case study of real-time WSN data," *Sensors*, vol. 19, no. 1, p. 166, Jan. 2019.
[10] D. F. Silva, R. Giusti, E. Keogh, and G. E. A. P. A. Batista, "Speeding up similarity search under dynamic time warping by pruning unpromising alignments," *Data Mining Knowl. Discovery*, vol. 32, no. 4, pp. 988–1016, Jul. 2018.
[11] J. Fan, Q. Sun, W.-X. Zhou, and Z. Zhu, *Principal Component Analysis for Big Data*. Hoboken, NJ, USA: Wiley, pp. 1–13, 2018.
[12] Y. Aït-Sahalia and D. Xiu, "Principal component analysis of high-frequency data," *J. Amer. Stat. Assoc.*, vol. 114, no. 525, pp. 287–303, 2019.
[13] C. Happ and S. Greven, "Multivariate functional principal component analysis for data observed on different (dimensional) domains," *J. Amer. Stat. Assoc.*, vol. 113, no. 522, pp. 649–659, Apr. 2018.
[14] L. Breiman, *Classification Regression Trees*. Evanston, IL, USA: Routledge, 2017.
[15] C. Zhang, F. Masseglia, and Y. Lechevallier, "The anti-bouncing data stream model for web usage streams with intralinkings," *Inf. Sci.*, vol. 278, pp. 757–772, Sep. 2014.
[16] C. Blum and M. J. Blesa, "Hybrid techniques based on solving reduced problem instances for a longest common subsequence problem," *Appl. Soft Comput.*, vol. 62, pp. 15–28, Jan. 2018.
[17] K.-T. Tseng, D.-S. Chan, C.-B. Yang, and S.-F. Lo, "Efficient merged longest common subsequence algorithms for similar sequences," *Theor. Comput. Sci.*, vol. 708, pp. 75–90, Jan. 2018.
[18] K. Bringmann and M. Künnemann, "Multivariate fine-grained complexity of longest common subsequence," in *Proc. 29th Annu. ACM-SIAM Symp. Discrete Algorithms*, 2018, pp. 1216–1235.
[19] C. Blum and M. J. Blesa, "A comprehensive comparison of metaheuristics for the repetition-free longest common subsequence problem," *J. Heuristics*, vol. 24, no. 3, pp. 551–579, Jun. 2018.
[20] B. Chiu, E. Keogh, and S. Lonardi, "Probabilistic discovery of time series motifs," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2003, pp. 493–498 .
[21] A. Mueen, E. Keogh, Q. Zhu, S. Cash, and B. Westover, "Exact discovery of time series motifs," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2009, pp. 473–484 .
[22] Y. Gao and J. Lin, "HIME: Discovering variable-length motifs in large-scale time series," *Knowl. Inf. Syst.*, vol. 61, no. 1, pp. 513–542, Oct. 2019.
[23] A. Mueen and N. Chavoshi, "Enumeration of time series motifs of all lengths," *Knowl. Inf. Syst.*, vol. 45, no. 1, pp. 105–132, Oct. 2015.
[24] B. Liu, J. Li, C. Chen, W. Tan, Q. Chen, and M. Zhou, "Efficient motif discovery for large-scale time series in healthcare," *IEEE Trans. Ind. Inform.*, vol. 11, no. 3, pp. 583–590, Jun. 2015.
[25] L. Xun and L. Zhishu, "The similarity of multivariate time series and its application," in *Proc. Int. Conf. Manage. e-Commerce e-Government*, Oct. 2010, pp. 76–81 .
[26] C. Zhang, Y. Hao, M. Mazuran, C. Zaniolo, H. Mousavi, and F. Masseglia, "Mining frequent itemsets over tuple-evolving data streams," in *Proc. 28th Annu. ACM Symp. Appl. Comput. (SAC)*, 2013, pp. 267–274 .
[27] G. Benson, A. Levy, S. Maimoni, D. Noifeld, and B. Shalom, "LCSk: A refined similarity measure," *Theor. Comput. Sci.*, vol. 638, pp. 11–26, Jul. 2016.
[28] D. Zhu, L. Wang, T. Wang, and X. Wang, "A space efficient algorithm for the longest common subsequence in k-length substrings," *Theor. Comput. Sci.*, vol. 687, pp. 79–92, 2017.
[29] Y. Ueki, M. Kurihara, Y. Matsuoka, K. Narisawa, R. Yoshinaka, H. Bannai, S. Inenaga, and A. Shinohara, "Longest common subsequence in at least k length order-isomorphic substrings," in *Proc. 43rd Int. Conf. Current Trends Theory Pract. Comput. Sci.*, Limerick, Ireland, Jan. 2017, pp. 363–374.
[30] J. Thioulouse, S. Dray, A.-B. Dufour, A. Siberchicot, T. Jombart, and S. Pavoine, *Multivariate Analysis of Ecological Data*, 4th ed. New York, NY, USA: Springer-Verlag, 2018.
[31] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley Interdiscipl. Rev., Comput. Statist.*, vol. 2, no. 4, pp. 433–459, 2010.
[32] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases," *Trans. Database Syst.*, vol. 27, no. 2, pp. 188–228, Jun. 2002.
[33] K. Yang and C. Shahabi, "A PCA-based similarity measure for multivariate time series," in *Proc. 2nd ACM Int. Workshop Multimedia Databases (MMDB)*, 2004, pp. 65–74 .
[34] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, p. 11, 2011.

[35] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Proc. 18th Int. Conf. Data Eng.*, Jun. 2003, pp. 673–684 .
[36] C. Zhang, G. Almpanidis, F. Hasibi, and G. Fan, "Gridvoronoi: An efficient spatial index for nearest neighbor query processing," *IEEE Access*, vol. 7, pp. 120997–121014, 2019.
[37] A. Apostolico, "String editing and longest common subsequences," in *Handbook of Formal Languages*, G. Rozenberg and A. Salomaa, Eds. Berlin, Germany: Springer, 1997.
[38] F. Duchêne, C. Garbay, and V. Rialle, "Similarity measure for heterogeneous multivariate time-series," in *Proc. 12th Eur. Signal Process. Conf.*, Sep. 2004, pp. 1605–1608.
[39] Y. Sakurai, M. Yoshikawa, and C. Faloutsos, "FTW: Fast similarity search under the time warping distance," in *Proc. 24th ACM SIGMOD-SIGACT-SIGART Symp. Princ. Database Syst.*, 2005, pp. 326–337.
[40] T. Rakthanmanon, B. Campana, A. Mueen, G. Batista, B. Westover, Q. Zhu, J. Zakaria, and E. Keogh, "Searching and mining trillions of time series subsequences under dynamic time warping," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2012, pp. 262–270 .
[41] R. Khan, I. Ali, M. Zakarya, M. Ahmad, M. Imran, and M. Shoaib, "Technology-assisted decision support system for efficient water utilization: A real-time testbed for irrigation using wireless sensor networks," *IEEE Access*, vol. 6, pp. 25686–25697, 2018.
[42] D. L. Wheeler, "Database resources of the national center for biotechnology information," *Nucleic Acids Res.*, vol. 33, pp. D39–D45, Dec. 2004.
[43] K. Bache and M. Lichman, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2013. [Online]. Available: http://archive.ics.uci.edu/ml
[44] D. Dua and C. Graff, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California, Irvine, CA, USA, 2019. [Online]. Available: http://archive.ics.uci.edu/ml

**RAHIM KHAN** received the Ph.D. degree in computer science from the GIK Institute of Engineering Sciences and Technology, Swabi, Pakistan. He is currently an Assistant Professor with the Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan. His research interest includes the wireless sensor networks (WSNs) deployment, the Internet of Thing (IoT), routing protocols, outliers detection, techniques for congestion control, decision support system (DSS), vehicular ad-hoc networks, data analysis, and similarity measures.

**MUHAMMAD ZAKARYA** received the Ph.D. degree in computer science from the University of Surrey, Guildford, U.K. He is currently a Lecturer with the Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan. His research interests include cloud computing, mobile edge clouds, performance, energy efficiency, algorithms, and resource management. He has deep understanding of the theoretical computer science and data analysis. Furthermore, he also owns deep understanding of various statistical techniques which are, largely, used in applied research.

**AYAZ ALI KHAN** received the M.Phil. (M.S.) degree in computer science from the COMSATS Institute of Information Technology (CIIT), Islamabad, Pakistan. He is currently pursuing the Ph.D. degree with the Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan. His areas of research include energy-aware and performance-efficient scheduling, resource allocation, placement, and management, at datacenter level. Moreover, he has enough knowledge of distributed systems, optimization, game theory, and computer programming.

**IZAZ UR RAHMAN** received the Ph.D. degree in computer science from the Department of Electronic and Computer Engineering, Brunel University, London, U.K. He is currently an Assistant Professor with the Department of Computer Science, Abdul Wali Khan University, Mardan, Pakistan. He is teaching AI and fuzzy systems to graduate and undergraduate level students. His research interest includes power systems, optimization algorithms such as particle swarms, fuzzy logic, distributed systems, machine learning, and artificial intelligence.

**MOHD AMIRUDDIN ABD RAHMAN** received the B.Sc. degree in electrical engineering from Purdue University, West Lafayette, USA, the M.Sc. degree in sensor and instrumentation from Universiti Putra Malaysia, and the joint (co-tutelle) Ph.D. degree from the University of Sheffield, U.K., and Universiti Putra Malaysia. During his Ph.D. studies, he worked on developing new algorithms for multifloor indoor localization system and also proposing new approach in indoor localization system based on WLAN. He has also joined Nokia Bell Labs, Dublin, Ireland, as a Post-Graduate Research Intern to carried out research and working with research team on co-localization and tracking algorithm for WLAN networks. He has also served as an Invited Researcher with Bell Labs, Antwerp, Belgium, to apply the localization system for Bell-Labs Future-X day. He is currently a Senior Lecturer with the Department of Physics, Universiti Putra Malaysia. He also leads the research in indoor localization with the Radio Frequency and Microwave Research Group. His other research interests also include signal processing, pattern recognition/matching/prediction, machine learning algorithms, electromagnetics-related computation and modeling, and RF and microwave-based sensor systems.

**MUHAMMAD KHALIS ABDUL KARIM** received the B.Sc. degree (Hons) in imaging diagnostic and radiotherapy from Universiti Kebangsaan Malaysia (UKM), in 2008, the M.Sc. degree in physics from Universiti Teknologi Malaysia, in 2013, and the Ph.D. degree in physics from Universiti Teknologi Malaysia (UTM), in December 2016. He has expertise in medical physics particularly radiation dosimetry in medical imaging and has been appointed as an advisory member in several technical committee during his work. Aside from working in clinical environment, he also actively involved giving lecture in several workshops on radiation protection and nuclear security. To date, he has authored or coauthored 35 international journals (with impact factor ranging from 0.2 to 5.1), ten international proceedings and three national proceedings. He has reviewed 20 international journal articles from reputable and well-established publisher. His other research interests include medical imaging, radiation dosimeters, and image processing.

**MOHD SHAFIE MUSTAFA** received the Ph.D. degree from Universiti Putra Malaysia. He is currently a Senior Lecturer with the Department of Mathematics and an Associate Researcher with the Institute for Mathematical Research. His research interests include applications of statistical modeling, robust designs, and experimental designs in some industrial problems, such as food science and chemical engineering study. He also expands his research interest in problems of response surface methodology with the development of robust techniques for designed experiments. His other research interests include regression analysis, small area estimation, and machine learning algorithms.