An Efficient Memristor-based Distance Accelerator for Time Series Data Mining on Data Centers

Xiaowei Xu[1], Dewen Zeng[1], Wenyao Xu[2], Yiyu Shi[3], Yu Hu[1*]
[1] Huazhong University of Science and Technology, Wuhan, China, 430074
[2] The State University of New York at Buffalo, Buffalo, NY, 14260
[3] University of Notre Dame, South Bend, IN, 46656
[*] Corresponding author: bryanhu@hust.edu.cn

ABSTRACT

The rapid development of Internet-of-Things (IoT) is yielding a huge volume of time series data, the real-time mining of which becomes a major load for data centers. The computation bottleneck in time series data mining is the distance function, which has been tackled by various software optimization and hardware acceleration techniques recently. However, each of these techniques is only designed or optimized for a specific distance function. To address this problem, in this paper we propose an efficient and reconfigurable memristor-based distance accelerator for real-time and energy-efficient data mining with time series on data centers. Common circuit structure is extracted to save chip areas, and the circuit can be configured to any specific distance functions. Experimental results show that compared with existing works, our work has achieved a speedup of 3.5x-376x on performance and an improvement of 1-3 orders of magnitude on energy efficiency.

1. INTRODUCTION

Energy efficiency of data centers has been a primary focus in the past a few years due to their excessive power consumption. On the other hand, the load on data centers keeps increasing with the explosion of information technologies. It has been predicted that by 2020 a major portion of the load will come from internet-of-things (IoT), which will yield over 4.4 zettabyes (5.5×10^{21} Bytes) of time series data by 2020 [4]. These time series data are transmitted to data centers for real-time mining [1]. It is therefore of utmost interest to explore techniques that handle time series data in real-time with high energy efficiency.

Classification, clustering and frequency pattern mining are three main data mining tasks for time series [27]. The computational bottleneck of these methods is the calculation of distance function, which is used to evaluate the similarity of two time series. Distance functions have a relatively high complexity, yet all data mining tasks will invoke it a huge number of times. Thus, the calculation of distance functions consumes a large fraction of the data mining time. For example, research results show that the computation of distance function takes up to more than 99% of the runtime for subsequence similarity search task [24].

Recently, software optimization and hardware acceleration have been widely exploited for distance functions. Dynamic time warping (DTW) has been optimized with lower bound methods [24], field programmable gate array (FPGA) [25][30], graphics processing unit (GPU) [25] and application-specific integrated circuit (ASIC) [18]. Manhattan distance (MD) has been accelerated with GPU [8]. Longest common subsequence (LCS), Hausdorff distance (HauD) and Hamming distance (HamD) have also been accelerated by GPU [22] [14][29]. Edit distance (EdD) has been optimized on GPUs [9] and ASICs [26]. However, each data center handles a variety of applications which use different distance functions. For example, a Google data center needs to deal with healthcare [2] and smart city applications [7]. The former adopts HamD for iris authentication [29] and LCS for electrocardiogram (ECG) similarity [10], while the latter uses DTW for vehicle classification [31]. None of these existing works can work well in this scenario as they are optimized for a single distance function only. It remains an open problem in the literature how to design a reconfigurable and efficient accelerator that works for all the distance functions, which is of ultimate importance on data centers.

In this paper, we address this problem by putting forward a novel efficient and reconfigurable memristor-based distance accelerator for real-time and energy-efficient time series data mining on data centers. Particularly, we present a specific analog circuit design as a unified hardware that can be reconfigured for a set of distance functions (including DTW, LCS, HauD, EdD, HamD, MD). We extract the basic primitives to facilitate various distance functions to save chip area. An emerging device memristor is adopted in analog circuit design for configurable resistance. Experimental results show that compared with existing works, our work has achieved a speedup of 3.5x-376x on performance and an improvement of 1-3 orders of magnitude on energy efficiency.

The rest of the paper is structured as follows: Section 2 describes the background. Section 3 presents the distance accelerator architecture and circuit designs. Experiments and discussion are given in Section 4. The paper is concluded in Section 5.

2. BACKGROUND

In this section, the widely adopted six distance functions are introduced. DTW, LCS and EdD are dynamic programming methods, which can handle two sequences with different length, while HamD and MD only support sequences with the same length. HauD can also support two sequences with different length. Considering that in real applications the significance of each element is different, weight is introduced. Interested readers can refer to [23][12][6] [32][21] [19] for the weighted versions of DTW, LCS, MD, HamD, HauD, and EdD, respectively.

Suppose there are two sequences P and Q as follows:

$$P = \{P_1, P_2, ..., P_i, ...P_m\}, Q = \{Q_1, Q_2, ..., Q_j, ...P_n\}, \quad (1)$$

where m and n are the length of Q and P, respectively.

The procedure of DTW calculation is a dynamic programming based iterates process. Specifically, DTW is to calculate a shortest warping path between two sequences P and Q, which is derived as shown in Equation 2, where D is the cumulate distance in

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC '17, June 18-22, 2017, Austin, TX, USA
Copyright 2017 ACM 978-1-4503-4927-7/17/06
http://dx.doi.org/10.1145/3061639.3062200 ...$15.00.

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on October 19,2024 at 09:09:08 UTC from IEEE Xplore. Restrictions apply.

the warping path. w_{ij} is the weight, which equals to 1 for general DTW and to other values for weighted DTW. Smaller $DTW(P,Q)$ value corresponds to higher similarity. Usually the Sakoe-Chiba band [24] is adopted for DTW, and its constraint R restricts the warping path. DTW has been optimized with lower bound methods [24], field programmable gate array (FPGA) [25][30], graphics processing unit (GPU) [25] and application-specific integrated circuit (ASIC) [18].

$$\begin{gathered} D_{i,j} = w_{i,j}|P_i - Q_j| + min\{D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}\}; \\ D_{0,0} = 0;\ D_{0,j} = D_{i,0} = \infty;\ \ 1{\le}i{\le}n;\ 1{\le}j{\le}m; \\ DTW(P,Q) = D_{n,m}. \end{gathered} \tag{2}$$

LCS is to find the longest common subsequence of two strings. In order to apply LCS to time series, *threshold* is introduced to determine whether two elements are equal or not. LCS is also belong to dynamic programming as shown in Equation 3, where V_{step} is the contribution of two equal elements. It should be noted that unlike DTW, smaller $LCS(P,Q)$ value corresponds to lower similarity. LCS has been accelerated by GPU [22].

$$\begin{gathered} L_{i,j} = \begin{cases} 0, \quad\quad if\ \ i=0\ \ or\ \ j=0 \\ L_{i-1,j-1} + w_{i,j}V_{step}, \\ \quad if\ \ i,j>0\ \ and\ \ |P_i - Q_j| \le threshold \\ max(L_{i,j-1}, L_{i-1,j}), \\ \quad if\ \ i,j>0\ \ and\ \ |P_i - Q_j| > threshold \end{cases} \\ LCS(P,Q) = L_{n,m}. \end{gathered} \tag{3}$$

EdD is the number of operations in individual characters to transform one string into another. Thus, lower EdD value means higher similarity. The permitted operations include *replacement*, *insertion* and *deletion*. By introducing *threshold*, EdD can also handle time series as shown in Equation 4. EdD has been optimized on GPUs [9] and ASICs [26].

$$\begin{gathered} E_{i,j} = \begin{cases} min(E_{i-1,j} + w_{i-1,j}V_{step}, E_{i,j-1} + w_{i,j-1}V_{step}, \\ \quad E_{i-1,j-1} + w_{i-1,j-1}V_{step})\ \ if\ \ |P_i - Q_j| \le threshold \\ min(E_{i-1,j} + w_{i-1,j}V_{step}, E_{i,j-1} + w_{i,j-1}V_{step}, \\ \quad E_{i-1,j-1}) \quad\quad\quad\quad\quad\quad if\ \ |P_i - Q_j| > threshold \end{cases} \\ E_{i,0} = i,\ \ E_{0,j} = j,\ \ EdD(P,Q) = E_{n,m}. \end{gathered} \tag{4}$$

HauD measures how far two subsets are from each other. Low HauD value means two sets are close (high similarity) or each point in one set is close to each point in another set. The computation of HauD is shown in Equation 5. HauD has been accelerated by GPU [14].

$$HauD = max_{j\in n}(min_{j\in n} w_{i,j}|P_i - Q_j|) \tag{5}$$

Hamming distance is the number of positions at which the corresponding characters are different. Like LCS and EdD, *threshold* is adopted for time series. The calculation process is shown in Equation 6. HamD has been accelerated by GPU [29].

$$\begin{gathered} H_i = \begin{cases} H_{i-1}\ \ if\ \ |P_i - Q_i| \le threshold \\ H_{i-1} + w_i V_{step}\ if\ \ |P_i - Q_j| > threshold \end{cases} \\ H_0 = 0, n = m,\ \ HamD(P,Q) = H_n. \end{gathered} \tag{6}$$

MD is a simple but rather popular method for time series [8], which is the sum of absolute differences in the corresponding positions. The calculation process is as shown in Equation 7. MD has been accelerated with GPU [8].

$$MD(P,Q) = \sum_{i}^{n} w_i|P_i - Q_i|,\ \ n=m. \tag{7}$$

From the above discussion it is clear that any existing accelerator is for a specific distance function only, and cannot be shared by multiple functions. However, this is exactly what is needed on data centers.

3. ACCELERATOR ARCHITECTURE

3.1 Architecture Overview

The proposed accelerator architecture comprises four modules: a Digital-to-Analog convertor (DAC) array, a computation module, a control and configuration module, and an Analog-to-Digital convertor (ADC) array as shown in Fig. 1. The DAC and ADC arrays are used to convert time series data between digital signals and analog signals. The control and configuration module has two responsibilities: 1) control the dataflow between modules; 2) reconfigure circuit connections in the computation module to perform specific distance functions with the configuration lib.

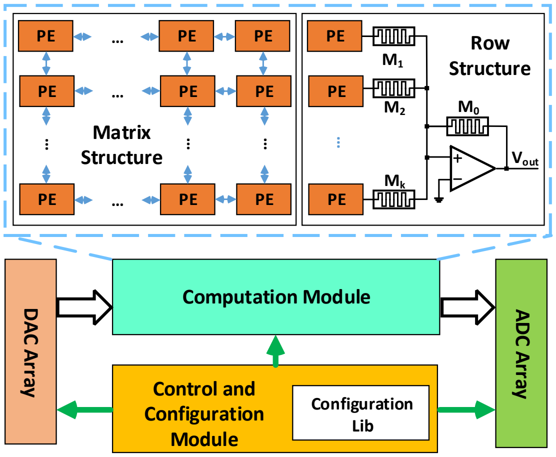
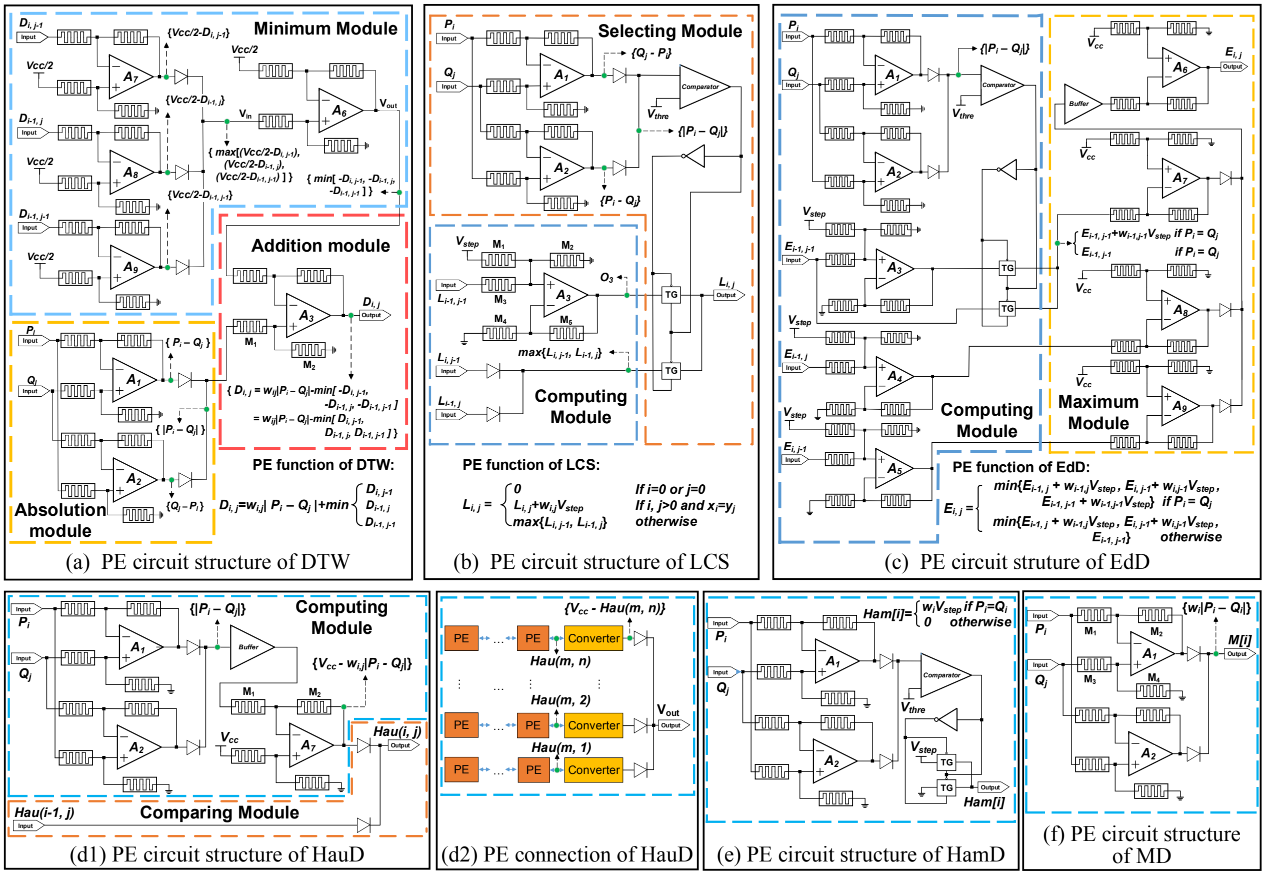


Figure 1: Architecture of the distance accelerator.

The configurable computation module calculates the distance functions. In order to save chip areas, we extract the basis primitive, the processing element (PE) of the analog circuits of distance functions. Each PE is compromised of nine analog subtracters, two transmit gates (TG), five diodes, one comparator, one buffer, and one convertor. The connections between the basic elements in PE is realized with TGs. All the adopted six distance functions are aggregated into two structures for the connection between PEs: matrix structure (for DTW, LCS, HauD and EdD) and row structure (for MD and HamD) as shown in Fig. 1. The circuit structures for different algorithms have a high similarity with each other in matrix and row structure, respectively. The reuse of op-amps and their corresponding memristors are labeled as shown in Fig. 2. It can be noticed that the configuration of connections for the two structures is relatively simple, and the circuit elements have a high resources utilization. By configuring each PE and connections between PEs, the function of specific distance can be achieved. The details of configurations are discussed in Section 3.2. When the sequence length is larger than the number of PEs in each row or column, tiling technique will be applied and the throughput will decrease.

In analog circuits, memristor is used for computation due to two reasons. First, using memristors as normal resistors enables the fine-tuning of memristance, which helps mitigate the impact of process variation and parasitic resistance. Secondly, By setting memristors to specific resistance, computation can be realized. A typical calculation of memristors is shown in the row structure in Fig. 1. V_{out} is the weighted sum of the output of each PE, and the weight is determined by the ratio of M_i ($1 \le i \le k$) and M_0. For general computation of MD, DTW, LCS, HamD, EdD, and HauD, the ratio of 1 is adopted, and only the high resistance state (HRS) and low resistance state (LRS) of memristors are used. Recently, weighted version of MD [23], DTW [12], LCS [6], HamD [32], EdD [21] and HauD [19] have been widely adopted for a variety of applications. In this situation, different ratios between memristors are used, and memristors need to be set to specific resistance other than HRS or LRS. The calculation with memristors in the matrix structure follows the same principle. Within analog circuits, the computation is conducted in a parallel manner. We discover that with identical circuit structure, the relations of outputs in convergence state and unconvergence state are the same, which could be used for further optimization. The details of implementations are discussed in Section 3.3.


Minimum Module
Addition module
Absolution module
PE function of DTW:
$D_{i,j}=w_{i,j}|P_i - Q_j|+min\{D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}\}$
$\{D_{i,j} = w_{ij}|P_i - Q_j|-min[-D_{i,j-1}, -D_{i-1,j}, -D_{i-1,j-1}]$
$= w_{ij}|P_i - Q_j|-min[D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}]\}$
$\{max[(Vcc/2-D_{i,j-1}), (Vcc/2-D_{i-1,j}), (Vcc/2-D_{i-1,j-1})]\}$
$\{min[-D_{i,j-1}, -D_{i-1,j}, -D_{i-1,j-1}]\}$
$\{|P_i - Q_j|\}$
$\{P_i - Q_j\}$
$\{Q_j - P_i\}$
V_{in}
V_{out}
Vcc/2
Input
Output
$D_{i,j}$
M_1
M_2
A_1 A_2 A_3 A_6 A_7 A_8 A_9
Selecting Module
Computing Module
Comparator
V_{thre}
V_{step}
M_1 M_2 M_3 M_4 M_5
O_3
TG
$L_{i,j}$
$max\{L_{i,j-1}, L_{i-1,j}\}$
PE function of LCS:
$L_{i,j} = \begin{cases} 0 & \text{If i=0 or j=0} \\ L_{i,j}+w_{i,j}V_{step} & \text{If i, j>0 and } x_i=y_j \\ max\{L_{i,j-1}, L_{i-1,j}\} & \text{otherwise} \end{cases}$
Maximum Module
Buffer
V_{cc}
$E_{i,j}$
$\{E_{i-1,j-1}+w_{i-1,j-1}V_{step}$ if $P_i = Q_j$
$E_{i-1,j-1}$ if $P_i = Q_j$
PE function of EdD:
$E_{i,j} = \begin{cases} min\{E_{i-1,j} + w_{i-1,j}V_{step}, E_{i,j-1}+ w_{i,j-1}V_{step}, E_{i-1,j-1} + w_{i-1,j-1}V_{step}\} & \text{if } P_i = Q_j \\ min\{E_{i-1,j} + w_{i-1,j}V_{step}, E_{i,j-1}+ w_{i,j-1}V_{step}, E_{i-1,j-1}\} & \text{otherwise} \end{cases}$
Comparing Module
$\{V_{cc} - w_{ij}|P_i - Q_j|\}$
Hau(i, j)
Hau(i-1, j)
PE
Converter
$\{V_{cc}$ - Hau(m, n)$\}$
Hau(m, n)
Hau(m, 2)
Hau(m, 1)
V_{out}
$Ham[i]=\begin{cases} w_iV_{step} & \text{if } P_i=Q_i \\ 0 & \text{otherwise} \end{cases}$
Ham[i]
$\{w_i|P_i - Q_i|\}$
M[i]

(a) PE circuit structure of DTW

(b) PE circuit structure of LCS

(c) PE circuit struture of EdD

(d1) PE circuit structure of HauD

(d2) PE connection of HauD

(e) PE circuit structure of HamD

(f) PE circuit structure of MD

Figure 2: PE circuit structures of DTW, LCS, EdD, HauD, HamD and MD. Particularly, HauD has a different PE connection.

3.2 Hardware implementation

3.2.1 Circuit of Dynamic Time Warping

The DTW calculation module is shown in Fig. 2(a), which includes three modules: absolution module, minimum module, and addition module. The absolution module calculates the absolute value of $(P_i - Q_j)$. Two analog subtractors are used for calculating $(P_i - Q_j)$ and $(Q_j - P_i)$, respectively. Two diodes are to output the larger value of the two values. Thus, the output value is the positive value, which is the absolute value of $(P_i - Q_j)$. For conditions of $P_i = Q_j$, the output is also correct. Weight factor $w_{i,j}$ supports weighted DTW, which can be achieved by configure memristors M_1 and M_2 to $M_1/M_2 = (2 - w_{i,j})/w_{i,j}$. Other memristors are all with the same resistance.

$$\begin{aligned} D_{i,j} &= w_{i,j}|P_i - Q_j| + min(D_{i,j-1}, D_{i-1,j}, D_{i-1,j-1}) \\ &= w_{i,j}|P_i - Q_j| + \{V_{cc}/2 - max(V_{cc} - D_{i,j-1}, \\ &\qquad V_{cc}/2 - D_{i,j-1}, V_{cc}/2 - D_{i,j-1})\} \quad Step\ 1 \\ &= w_{i,j}|P_i - Q_j| - \{max(V_{cc}/2 - D_{i,j-1}, V_{cc}/2 \\ &\qquad -D_{i,j-1}, V_{cc}/2 - D_{i,j-1}) - V_{cc}/2\} \quad Step\ 2 \end{aligned} \tag{8}$$

The minimum module obtains the minimum value of $D_{i,j-1}$, $D_{i-1,j}$, and $D_{i-1,j-1}$. As diodes are perfect for maximum value calculation, we transform the minimum calculation to a maximum problem as shown in Equation (8), where V_{cc} is the supply voltage. In Step 1, the minimum problem is converted to a maximum problem, which can be easily calculated with diodes. However, there is a problem in the designs according to Step 1. With diodes, the input current for the analog subtracter is fixed to positive, which means there is no negative current. As a result, the diode works in the cutoff region when the input is less than $V_{cc}/4$, and there is no current for the input. Thus, the maximum value for the output is $V_{cc}/4$, which is insufficient for DTW calculation. Step 2 is introduced to tackle the problem. The input and $V_{cc}/2$ switch their roles as shown in Fig. 2(a). Then, the output is the minimum value with a negative sign, which can be easily solved by converting addition to substraction.

3.2.2 Circuit of Longest Common Subsequence

The PE circuit of LCS is shown in Fig. 2(b). The calculation of $L_{i,j}$ depends on the elements of sequences and PEs besides it.

The PE circuit contains two modules: a selecting module and a computing module. The selecting module fulfills the calculation of conditions in Equation (3). To determine whether P_i is equal to Q_j, we first calculate the absolute value of $(P_i - Q_j)$, and then compare the absolute value with a threshold voltage V_{thre}. If the absolute value is less than the threshold voltage, we assume that P_i is equal to Q_j, otherwise not. The transmission gates (TG) determines which part should connect to the output.

The computing module is consisted of two parts. The first part calculates the sum of $L_{i-1,j-1}$ and $w_{i,j}V_{step}$. The second part outputs the maximum value of $L_{i,j-1}$ and $L_{i-1,j}$ with diodes. Weight factor $w_{i,j}$ supports weighted LCS by configuring memristors M_1, M_2, M_3, M_4 and M_5. Assuming $M_1/M_2 = k_1$, M_3 should be set to $w_{i,j}k_1M_2$, and the relation of M_4 and M_5 is $M_5/M_4 = (1 + k_1)w_{i,j}$.

3.2.3 Circuit of Edit Distance

Fig. 2(c) shows that the PE circuit of EdD includes two modules: a computing module and a minimum module. In the computing module, we have three computation paths. The first computation path is associated with $E_{i-1,j-1}$, which is the result of the left-lower PE. We calculate the absolute value of $(P_i - Q_j)$ and use a comparator to determine whether P_i is equal to Q_j. If P_i is equal to Q_j, the output of the comparator will be high and the output of the first path will be $E_{i-1,j-1} + w_{i-1,j-1}V_{step}$, otherwise will be $E_{i-1,j-1}$. The second and the third path share the same circuit structure, and the outputs are $E_{i-1,j} + w_{i-1,j}V_{step}$ and $E_{i,j-1} +$

$w_{i,j-1}V_{step}$, respectively. V_{step} is a unit voltage, and the exact result can be obtained by dividing $E(m, n)$ by V_{step}. For weighted LCS the configuration of memristors around op-amp A_3, A_4 and A_5 in Fig. 2(c) are the same with that in Fig. 2(b).

The minimum module calculates the minimum value among the output of the three paths in the computing module. As the diodes can easily solve the maximum problem, we use a subtractor circuit to make it a maximum problem.

The same problem arises here, which also exists in the PE circuit structure of DTW. The current through the diode must be in the right direction, which means the output of the diodes in the maximum module must be higher than $V_{cc}/2$. In order to solve the problem, we add a buffer at the output of the diodes to insure that the output can be lower than $V_{cc}/2$.

3.2.4 Circuit of Hausdorff Distance

Fig. 2(d1) shows the PE circuit structure of HauD, which is compromised of a computing module and a comparing module. The computing module is consisted of two steps, the first step is to calculate the absolute value of (P_i-Q_j). As explained in Section 3.2.1, diodes and V_{cc} are also used here to solve the minimum problem in the second step.

The comparing module outputs the maximum value of $D(i-1, j)$ and $V_{cc} - w_{i,j}|P_i - Q_j|$. We add a buffer between the output of diodes and the negative input of A_3 (shown in Fig. 2(d1)), therefore the output voltage of $w_{i,j}|P_i - Q_j|$ can be below V_{cc}/2. For weighted HauD, the configuration of memristors $M_2/M_1 = M_3/M_4 = w_{i,j}$ should be applied.

Fig. 2(d2) shows the PE circuit structure of HauD. Given Q_j, we check every elements of sequence P and calculate the value of $Hau(m, j)$, which is the maximum value of $V_{cc} - w_{i,j}|P_i - Q_j|$ ($1 \leq i \leq k$). With the same processing for Q_j in sequence Q, we have $Hau(m, 1)$, $Hau(m, 2)$,..., $Hau(m, n)$. Then, a converter is used to process each $Hau(m, j)$ in which the output is the difference of V_{cc} and $Hau(m, j)$. Therefore, the output of the converter is the minimal $w_{i,j}|P_i - Q_j|$ where j is fixed and i varies. Finally, we use diodes to output the maximum value of all minimal $w_{i,j}|P_i - Q_j|$, and the result is the HauD of P and Q.

3.2.5 Circuit of Hamming Distance

The PE circuit structure of HamD is shown in Fig. 2(e). The absolute value calculation module and a comparator are used to calculate whether P_i is equal to Q_j. If P_i is equal to Q_j, the output of the comparator will be high, and the output of $Ham[i]$ will be V_{step}. Otherwise, the output will connect to the ground, and $Ham[i]$ will remain zero. When all PEs finish computation, an analog adder is adopted to add all $Ham[i]$, and the output is the HamD of P and Q. Weighted HamD is achieved by configuring memristors to $M_0/M_k = w_k$ in the row structure in Fig. 1.

3.2.6 Circuit of Manhattan Distance

Fig. 2(f) shows the PE circuit structure of MD, which is the subset of that of HamD. Like HamD, when all the PE fulfill computation, we use an analog adder to add all $D[i]$, and the output is the MD of P and Q. For weighted MD, the configuration is the same with weighted HamD.

3.3 Implementation Details

(1) **Optimization**: In the row structure, each input has an equal position to each other, and the circuit structure for each input is identical. With this character, early decision can be achieved, which means HamD and MD can process sequences with a shorter time rather than the convergence time. The detail is illustrated with MD in Fig. 3. It can be noted that the relation of $|V(MD_1)|$, $|V(MD_2)|$ and $|V(MD_3)|$ in the unconvergence state and the convergence state are the same. This feature in analog domain is extremely useful for many data mining tasks. For example, in classification we can obtain the value at the *Early Point* shown in Fig. 3. The sequence with the minimum value obtained at the *Early Point* is also the one with the minimum value obtained in the convergence state.

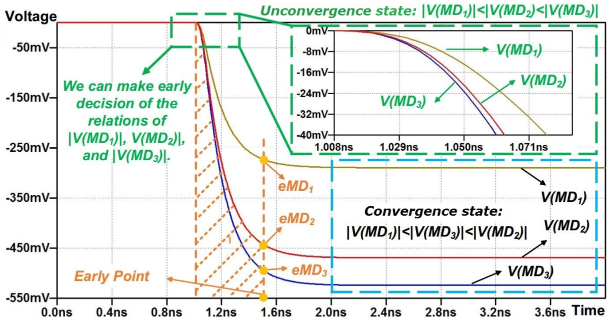


Figure 3: Early determination in analog circuits.

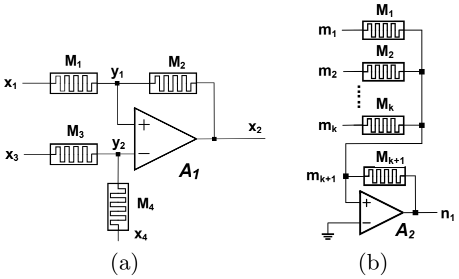


Figure 4: Resistance tuning circuit: (a) analog subtractor and (b) analog adder.

(2) **Resistance Tuning**: All the resistances in the distance accelerator are memristors. Thus, resistance tuning is required to make appropriate configurations for efficient computation [16]. This is also useful to minimize the influence of parasitic resistance. The process is presented as follows, which includes two parts, analog subtractor and analog adder as shown in Fig. 4.

For analog subtractors as shown in Fig. 4(a), we set $y_1 = 0$ and $y_2 = 0$ in the first step. The four ports, x_1, x_2, x_3 and x_4 are used to modulate M_1, M_2, M_3 and M_4, respectively. In the second step, we verify the ratio of M_1/M_2 and M_3/M_4. When verifying M_1/M_2, we set $y_2 = 0$ and $x_1 = 0.1$. By measuring x_2, the radio of M_1/M_2 can be verified. For example, for analog subtractors in LBE, M_1 and M_2 are set to HRS. Thus, if $x_2 = 0.1V$, $M_1/M_2 = 1$ is configured successfully. When verifying M_3/M_4, we set $x_3 = 0.1V$ and $x_4 = 0$. By measuring y_2, the radio of M_3/M_4 can be verified. If verification is not successful, the first step will be applied to further modulate corresponding memristors. The two steps can be iterated several times for better precision.

For analog adders as shown in Fig. 4(b), we set $n_2 = 0$ in the first step. The $k + 1$ ports, m_1, m_2,..., m_k and m_{k+1} are adopted to modulate M_1, M_2,..., M_k and M_{k+1}, respectively. In the second step, M_{k+1} is regarded as the reference memristor, which is used to verify other memristors. We will set $m_1 = 0.1V$ and measure n_1 to verify M_1/M_{k+1}. If $n_1 = 0.1V$, the configuration of $M_1 = M_{k+1}$ is achieved. Otherwise, M_1 will be modulated according to the offset to the configuration. The process of modulation and verification can be iterated for high precision. The above tuning process for M_1 will be applied to other memristors.

(3) **Impact of Process Variation**: Considering process variation, the actual resistance of memristors have a tolerances of ±20% to ±30%, which will degrade the solution quality. Two steps are adopted to reduce the impact of process variation. Firstly, we can discover that the solution quality is only the ratio of memristors. Thus, tolerance control technique [11] can be used to restrict the tolerance between two memristors lower than 1%. Secondly, post-fabrication resistance tuning can further reduce the negative effects of process variation.

4. EXPERIMENTAL RESULTS

4.1 Experimental Setup

We adopt three data sets (Beef, Symbols, and OSU Leaf) from the UCR Time Series Classification Archive [13]. For each data set, we formalize the sequences with different lengths.

We implement the proposed design in SPICE [20] with the 32nm technology node, and the simulation setup is presented in TABLE 1. It should be noted that we focus on the computation part in the simulation, and weights are set to 1 to make a fair comparison with existing works. It should be highlighted that different weights have little influence on the performance. For the sake of generality, the parameters of op-amps and diodes are set to typical values according to recent literatures [17]. Particularly, a parasitic capacitance of 20fF is added to each circuit net to model the effect of parasitic capacitance [17]. The parameter voltage resolution is to translate sequence values to voltages. As the runtime time is about 20 hours for DTW simulations for sequences of length 40, the longest sequence length is set to 40. Considering sequence length, we set the voltage resolution to 20mV. The translation is as follows: the sequence value 1 is translated to 20mV. Other values follow the same principle, e.g., 1.2 and -0.5 are translated to 24mV and -10mV, respectively. The stochastic Biolek's model [5] considering nondeterministic digital dynamics for memristor simulation is adopted, and the parameters are shown in TABLE 2.

For algorithms such as EdD, LCS and HamD, a threshold voltage (V_{thre}) and a unit voltage (V_{step}) are used. Considering the longest sequence length is 40, we set V_{step} to 10mV in case the output voltage overflows. Unlike V_{step}, V_{thre} is application-specific.

Table 1: SPICE parameters for distance accelerator setup

Parameters	Configuration
Open loop gain of op-amp	1×10^4
Gain-bandwidth product of op-amp (GHz)	50
V_{cc}(V)	1.0
Voltage resolution	20mV for 1
Threshold voltage of diodes (V)	0 [17]

Table 2: Parameters for Stochastic Biolek's model

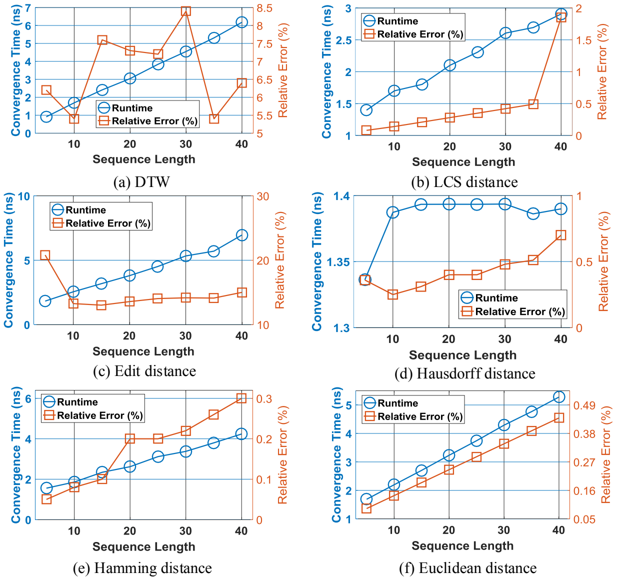
V_0	τ	V_{T_0}	ΔV	R_{off}	R_{on}	$\Delta R_{on/off}$
0.156V	2.85×10^5s	3.0V	0.2V	100kΩ	1kΩ	5%

4.2 Results and Analysis

We present performance evaluation for each module of these algorithms. The convergence time indicating how fast the module can operate and the relative error are discussed. The convergence time is defined as the interval between the rising edge of the input and the timestamp when the output is within 0.1% of the final value. For each algorithm module, we randomly choose a pair of data from the same class and a pair from different classes in one dataset. The length of the time series data are converted to different lengths. Totally 10 similarity computations are presented for each dataset. This process is repeated for all the three datasets.

The convergence time and relative error of the six distance functions is shown in Fig. 5. We can observe that the convergence time for all distance functions are almost linear to the sequence length except for HauD. This linearity is due to the fact that the current propagation path of all the distance functions expect HauD have a linear capacitance to the input size. It can be noticed that the relative error of DTW and EdD is larger than others', which is caused by the fact that larger zero drift exists PEs for DTW and EdD.

We can discover that the convergence time of HauD stays almost constant when the sequence length is larger than 10. This is because the convergence time is determined by the output voltage and the amount of capacitance in the current propagation path. Actually, the efficiency of capacitance increasing works only when the output voltage increases. For HauD, it should be noted that the result of each sub-module is only used for the maximum calculation in the sub-module right to it, whose calculation time is very short and can be ignored compared to other calculation. Thus, these sub-modules work almost in parallel, and the increase of sequence length has almost no affect on the runtime. With the fact that the output voltage of HauD will not increase when the sequence length increase, the convergence time of HauD stays constant basically.



(a) DTW
(b) LCS distance
(c) Edit distance
(d) Hausdorff distance
(e) Hamming distance
(f) Euclidean distance

Figure 5: Convergence time and relative error of distance functions.

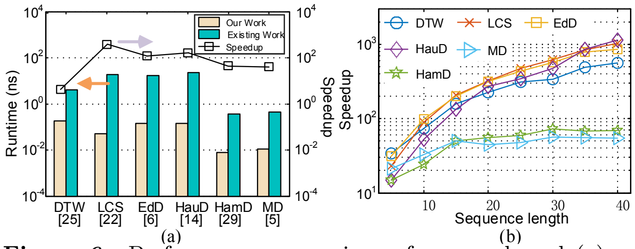


Figure 6: Performance comparison of our work and (a) existing works and (b) CPU implementation.

It should be highlighted that the relative error of HamD and MD are linear to the sequence length. This is because each sub-module of these two algorithms is attached with a fixed small absolute error. When the sequence length increases, the small error is added to the final relative result linearly. The error can be regarded as a bias, which has no significant influence on the relation of results.

In the module performance experiment, all the results are not influenced by the nondeterminism of the stochastic Biolek's model. This is due to the following two reasons. Firstly, all memristors are under a voltage far less than the threshold voltage of memristors. It should be noted that only in the sub-threshold voltage, it is probabilistic to form a single filament for stochastic resistance change. For DTW, the input voltages in the absolution module are very small, which is far lower than the threshold voltage of 3.0V. In the minimum module, the output voltage of diodes cannot be below zero, which makes the input voltages have a value less than or equal to $V_{cc}/2$. Thus, the voltage drop of all the memristors in the minimum module and the addition module is less than or equal to $V_{cc}/4 = 0.25V$, which is also far lower than the threshold voltage of 3.0V. Other distance functions has the same situations. Secondly, the computation time is far less than the transition time of about $1\mu s$ for memrisotrs. However, the running time for distance functions is about several nanoseconds. Considering the above two conditions, the possibility for stochastic resistance change is rather low with several hundreds of experiments.

4.3 Performance Comparison

The performance comparison of our work and existing works [25][22][9][14][29][8] is shown in Fig. 6(a). As all existing hardware accelerations and our work have a linear time complexity of the sequence length, the processing time of each element in sequences is analyzed for speedup discussion. For HamD and MD, the optimization method early determination is adopted, and the point with one-tenth convergence time is set as $Early\ Point$. We can notice that our work has a speedup of 3.5x-376x for the six distance functions. The runtime of LCS and HamD in our work is shorter than that of others. This is because the convergence time in analog circuits is influenced by output voltages which are smaller for LCS and HamD.

As existing works have different configurations for different applications, we also make an appropriate comparison of our work and a CPU implementation with the same datasets. The desktop computer is with Windows 8.1 operation system and a quad-core i5-3470 CPU. The code is written in C language and compiled by Microsoft Visual Studio 2015. The optimization level is set to maximum speed $O2$. As shown in Fig. 6(b), our work has a speedup of 20x-1000x compared to CPU with different sequence lengths. The speedup gets larger with longer sequences. It should be noted that the speedup for HamD and MD are smaller than the other four distance functions. This is because that the time complexity of the two distance functions is $O(n)$, while that of others are $O(n^2)$.

A rough power analysis is presented for energy efficiency discussion. The power for a recently popular op-amp with a gain-bandwidth product 303GHz is $197\mu W$ [33] under $0.35\mu m$ technology node, and the power for the 32nm technology node is projected to $18\mu W$ with ideal scaling for capacitance. The same procedure goes for a recent 8-bit 1.6 Gsample/s DAC [28] in 90nm technology node, and the projected power for the adopted DAC is $32mW$. A recent $35mW$ 8.8 GSample/s ADC in 32nm technology node [15] is adopted. The number of PEs in each column and row is set to 128, which is the same with [25]. For sequence length larger than 128, tiling technique can be applied. For DTW configuration, the power consumption of the distance accelerator includes three parts: op-amps, ADCs/DACs, and memristors around op-amps. The widely-applied Sakoe-Chiba band constraint $R = 5\% \times n$ is adopted. The power consumption of the active op-amps is $(7R(2n - R)) \times 18\mu W = 0.20W$, while the power consumptions of DACs and ADCs are $\lceil Throughput_{in}/1.6GSample/s \rceil \times 32mW = 0.13W$ and $\lceil Throughput_{out}/8.8GSample/s \rceil \times 35\ mW = 0.026W$. Assuming at least one memristor is set to HRS from the source to the ground, the power consumption of memristors is $(7R(2n - R)) \times 2 \times 10\mu W = 0.22W$. Thus, the total energy consumption for DTW configuration is $0.58W$. Following the same principle, the total power consumptions of the distance accelerator for LCS, EdD, Haud, HamD, and MD are $2.97W$, $6.36W$, $2.64W$, $2.95W$, and $2.16W$, respectively. For the power consumption of the existing work, we use Xilinx Power Estimators [3] to estimate the power according to the used logical resources and clock frequency for FPGA implementations. For GPU implementations, we adopt 80% of the maximum power as the typical power. Thus, power consumptions of exiting work for DTW, LCS, EdD, Haud, HamD, and MD are $4.76W$ (FPGA), $240W$(GPU), $175W$(GPU), $120W$(GPU), $150W$(GPU), and $137W$ (GPU), respectively. Considering speedups, the improvement of energy efficiency is one to three orders of magnitudes (26.7x-8767x). Though more detailed implementation will weaken the speedup, the distance accelerator still has a higher energy efficiency.

5. CONCLUSIONS

In this paper, we propose an efficient and reconfigurable high-throughput memristor-based distance accelerator for time series data mining on data centers. We adopt memristors to design analog circuits for six widely-used distance functions including dynamic time warping, longest common subsequence, Hausdoff distance, edit distance, Hamming distance, and Manhattan distance. The basis primitive of the circuits is extracted, which can be configured to any specific distance functions. Compared with existing works, the performance of the proposed accelerator has a speedup of 3.5x-376x. Energy analysis shows that the accelerator has an improvement of 1-3 orders of magnitude on energy efficiency.

6. REFERENCES

[1] Gartner inc. Gartner Says the Internet of Things Will Transform the Data Center, 2014.

[2] Cisco inc. Cisco Data Center for Healthcare, 2016.

[3] Xilinx inc. https://www.xilinx.com, 2016.

[4] A. Adshead. Data set to grow 10-fold by 2020 as internet of things takes off. *ComputerWeekly. com*, 9, 2014.

[5] M. Al-Shedivat, R. Naous, et al. Memristors empower spiking neurons with stochasticity. *IEEE journal on ESTCS*, 2015.

[6] A. Banerjee and J. Ghosh. Clickstream clustering using weighted longest common subsequences. In *SIAM*.

[7] N. Z. Bawany and J. A. Shamsi. Smart city architecture: Vision and challenges.

[8] D.-J. Chang and et. al. Compute pairwise manhattan distance and pearson correlation coefficient of data points with gpu. In *SNPD'09*, pages 501–506. IEEE, 2009.

[9] R. Farivar and et. al. An algorithm for fast edit distance computation on gpus. In *InPar, 2012*, pages 1–9. IEEE, 2012.

[10] T. S. Han and et. al. Efficient subsequence matching using the longest common subsequence with a dual match index. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 2007.

[11] R. A. Hastings. *The art of analog layout*. Prentice Hall, 2006.

[12] Y.-S. Jeong and et. al. Weighted dynamic time warping for time series classification. *Pattern Recognition*.

[13] E. Keogh and T. Folias. The ucr time series data mining archive. *University of California, Riverside, CA. http://www.cs.ucr.edu/eamonn/TSDMA/index.html*, 2002.

[14] Y.-J. Kim, Y.-T. Oh, and et al. Precise hausdorff distance computation for planar freeform curves using biarcs and depth buffer. *The Visual Computer*, 26(6-8):1007–1016, 2010.

[15] L. Kull and et. al. A 35mw8 b 8.8 gs/s sar adc with low-power capacitive reference buffers in 32nm digital soi cmos. In *VLSI*, pages C260–C261. IEEE, 2013.

[16] B. Liu, M. Hu, et al. Digital-assisted noise-eliminating training for memristor crossbar-based analog neuromorphic computing engine. In *DAC*, pages 1–6. IEEE, 2013.

[17] G. Liu and Z. Zhang. A reconfigurable analog substrate for highly efficient maximum flow computation. In *DAC*, 2015.

[18] R. Lotfian and R. Jafari. An ultra-low power hardware accelerator architecture for wearable computers using dynamic time warping. In *DATE*, 2013.

[19] Y. Lu and et. al. An approach to word image matching based on weighted hausdorff distance. In *DAR*, 2001.

[20] L. W. Nagel and et. al. *SPICE: Simulation program with integrated circuit emphasis*. 1973.

[21] F. M. Oliveira-Neto and et. al. Online license plate matching procedures using license-plate recognition machines and new weighted edit distance. *TR part C*, 21(1):306–320, 2012.

[22] A. Ozsoy, A. Chauhan, and M. Swany. Fast longest common subsequence with general integer scoring support on gpus. In *PMAMM*, page 92. ACM, 2014.

[23] V. Perlibakas. Distance measures for pca-based face recognition. *Pattern Recognition Letters*.

[24] T. Rakthanmanon and et. al. Searching and mining trillions of time series subsequences under dynamic time warping. In *18th ACM SIGKDD*, pages 262–270. ACM, 2012.

[25] D. Sart, A. Mueen, and et al. Accelerating dynamic time warping subsequence search with gpus and fpgas. In *ICDE*, pages 1001–1006. IEEE, 2010.

[26] J. J. Tithi, N. C. Crago, and J. S. Emer. Exploiting spatial architectures for edit distance algorithms. In *ISPASS*, pages 23–34. IEEE, 2014.

[27] C.-W. Tsai and et. al. Data mining for internet of things: a survey. *IEEE Communications Surveys & Tutorials*, 2014.

[28] W.-H. Tseng and et. al. A cmos 8-bit 1.6-gs/s dac with digital random return-to-zero. *IEEE TCSII*, 58(1):1–5, 2011.

[29] N. A. Vandal and M. Savvides. Cuda accelerated iris template matching on graphics processing units. In *BTAS*. IEEE, 2010.

[30] Z. Wang, S. Huang, and et al. Accelerating subsequence similarity search based on dynamic time warping distance with fpga. In *FPGA*, pages 53–62. ACM, 2013.

[31] G. Weng and et. al. The vehicle's classification recognition system based on dtw algorithm. In *WCICA 2004*. IEEE, 2004.

[32] L. Zhang, Y. Zhang, J. Tang, K. Lu, and Q. Tian. Binary code ranking with weighted hamming distance. In *CVPR*, 2013.

[33] L. Zuo and S. K. Islam. Low-voltage bulk-driven operational amplifier with improved transconductance. *IEEE TCSI*, 2013.