Journal of Multidisciplinary Engineering Science Studies (JMESS)
ISSN: 2458-925X
Vol. 2 Issue 4, April - 2016

Improving Performance in Pattern Discovery

Modification Applied In Algorithm for Time Series

M. Sc. Nertila Ismailaja
Armundia Factory
Tirana, Albania
n.ismailaja@armundiafactory.com

***Abstract*—Time series analysis is a recent field of studies. It studies the impact that time has on regular data. Therefore, it is possible to determine a certain rule to describe the time series mathematically. On the other side, in order not to focus on a certain rule, but to depend on randomness, a time series is described by motifs, which are patterns repeated throughout the time series. In their articles, Mueen et al. ([1][2][3][4]) presented four approaches to discover motifs in time series. Dhamo et al [5][6] presented an algorithm to improve accuracy and quality in motif detection and also compared distance such as CID [7] with Chouakria index[8] with CID [5], where a better performance was given by Chouakria index with CID. A successive improvement of this algorithm was presented in their article by Lin et al[9], which aimed to present the importance of normalization of subsequences in time series as a preprocess. In this article, we aim to improve in ulterior the quality in pattern discovery, independently from the indexes in subsequences. The use of Chouakria index with CID as similarity measure is used to provide more satisfactory results than others, such as CID, Euclid, etc. The criteria used for comparison is execution time, number of motifs discovered and mean distance of similar subsequences from the 1-motif. In all cases, the modification to the algorithm provided the same/better performance than the previous algorithm presented by Dhamo[5][6]. The codes and plots are made in R.**

Keywords— Motif discovery; time series; algorithm; improvement; R

I. INTRODUCTION

Time series analysis is a wide field of studies that comprehends detecting features in time series, such as trend, periodicity, seasonality, etc, in order to fit a model that depends on time. On the other hand, researchers have pointed out that independently from values, times series' behavior tends to be repeated in small intervals (motifs). The first to introduce motifs in time series was Mueen et al. [1][2][3][4], who presented four approaches to discover motifs in time series, such as Online motif discovery, k-motif discovery, enumeration of motifs of all lengths and fixed length. Moreover, the detection for similarity in motifs was made non-trivial. Besides these methods, another important one is 1-motif, which finds the most repeated pattern throughout the time series. An enhancement in results was achieved by Lin et al.[10] who proposed normalization of subsequences in time series in order to eliminate the effect that values had in pattern discovery. In the following years, the aim was not only to find pattern, but also to propose similarity distances, so as to minimize the effect of approximate values. One of the most efficient similarity measures is CID, proposed by Batista et al[7]. Another important similarity measure is also Chouakria index[8], which is a product of Euclid distance and a coefficient. A combination of CID and Chouakria index was firstly proposed by Dhamo et al[6] in the research related to 1-motif. This similarity measure provided the best results according to a bigger number of similar patterns discovered in a time series, to a better quality but augmented the execution time due to its larger complexity. The algorithm used by Dhamo et al[5][6] has in basis brute-force algorithm, which expands its search in a one-by-one comparison.

II. TIME SERIES. MOTIF DISCOVERY ALGORITHM

A. Concepts and basic definitions in time series

Time series is an attempt to formalize specific phenomenon such as air temperature, water flows in a river, etc by gathering data regularly and elaborating them.

Definition 1 A time series T of length n is a set of data gathered in regular intervals $T = (T_1, T_2, \dots, T_n)$.

An example of a time series is given in Fig.1.

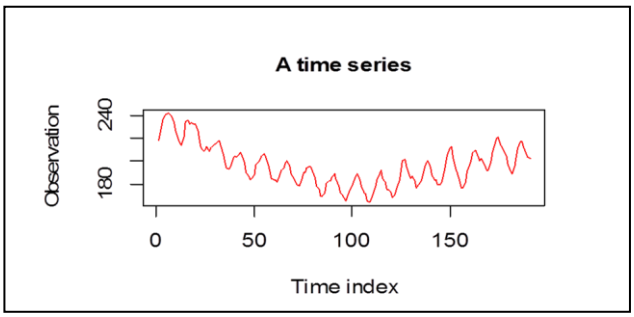

Fig.1. An example of a time series

Two main characteristics of a time series are mean and standard deviation, defined as follows:

Definition 2 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, its mean $\bar{T}$ is:

$$\bar{T} = \sum_{i=1}^{n} T_i \quad (1)$$

Definition 3 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, the standard deviation $sd(T)$ is:

$$sd(T) = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(T_i - \bar{T})^2} \quad (2)$$

It is understandable that both mean and standard deviation are numbers.

B. Motifs in time series - basic concepts

When trying to detect patterns in a time series, two principal indicators are the length of the pattern and the criterion used to compare whether two subsequences of a time series are or not similar and, moreover, the match should not be trivial.

Definition 4 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, a subsequence M_i of T with length m is defined as:

$$M_i = (T_{i+1}, T_{i+2}, \dots, T_{i+m-1}) \quad (3)$$

, where $i \geq n - m + 1$ and $m \in N$.

While comparing two subsequences M_i and M_j, to determine whether they are similar or not, distance is necessitated.

Definition 5 A distance is a function that complies with the following properties:

- Identity: $\forall x, y \in R, d(x, y) = 0 \leftrightarrow x = y$
- Symmetry: $\forall x, y \in R, d(x, y) = d(y, x)$
- Non-negativity: $\forall x, y \in R, d(x, y) \geq 0$
- Transitive: $\forall x, y \in R, \exists z \in R | d(x, y) \leq d(x, z) + d(z, y)$

The most utilized distance is Euclid, but, in contrary to its wide use, provides dissatisfactory results, because it can not afford time series' complexity. Therefore, a most common comparison mean is similarity measure.

Definition 6 Euclid distance between two subsequences with length m, $M_i = (T_{i+1}, T_{i+2}, \dots, T_{i+m-1})$ and $M_j = (T_{j+1}, T_{j+2}, \dots, T_{j+m-1})$ is the index, measured as below:

$$d_{Euc}(M_i, M_j) = \sum_{k=1}^{m}(M_{i,k} - M_{j,k})^2 \quad (4)$$

Definition 7 A similarity measure is a function that does not comply with one or more conditions to be a distance.

C. Similarity measures

In their studies, Dhamo et al[5][6] compared the efficiency of many similarity measures, such as Euclid, CID, Chouakria index with Cid or Euclid. Even though Batista first proposed CID in 2012, better results were provided by the use of Chouakria's temporal correlation coefficient (firstly proposed in 2008) in Chouakria's index. Better results were gained by a combination of both.

Definition 7 CID distance between two sub-sequences $M_i = (T_{i+1}, T_{i+2}, \dots, T_{i+m-1})$ and $M_j = (T_{j+1}, T_{j+2}, \dots, T_{j+m-1})$ with length m from the time series T, is the index, measured as:

$$d_{CID}(M_i, M_j) = d_{Eucl}(M_i, M_j)\frac{\max\{CE(M_i), CE(M_j)\}}{\min\{CE(M_i), CE(M_j)\}} \quad (5)$$

, where $CE(M_i)$ is the Complexity Estimation for the subsequence $M_i$and is defined as:

$$CE(M_i) = \sum_{k=1}^{m-1}(M_{i,k} - M_{i,k+1})^2 \quad (6)$$

Definition 8 Chouakria's index between two sub-sequences $M_i = (T_{i+1}, T_{i+2}, \dots, T_{i+m-1})$ and $M_j = (T_{j+1}, T_{j+2}, \dots, T_{j+m-1})$ with length m from the time series T, is the index which is measured as:

$$d_{CID}(M_i, M_j) = \frac{2}{1+e^{k*\delta(M_i, M_j)}} * COR_t(M_i, M_j) \quad (7)$$

, where $COR_t(M_i, M_j)$ is the temporal correlation coefficient, defined as:

$$COR_t(M_i, M_j) = \frac{\sum_{k=1}^{m-1}(M_{i,k} - M_{i,k+1})(M_{j,k} - M_{j,k+1})}{\sqrt{\sum_{k=1}^{m-1}(M_{i,k} - M_{i,k+1})^2}\sqrt{\sum_{k=1}^{m-1}(M_{j,k} - M_{j,k+1})^2}} \quad (8)$$

and $k \in R^+$; and $\delta(M_i, M_j)$ may be Euclid, etc. In their article, Dhamo et al. [6] proposed CID as similarity measure $\delta(M_i, M_j)$ and $k = 2$.

III. THE MOTIF DISCOVERY ALGORITHM. THE PROPOSED MODIFICATION

A. The Algorithm to detect similar patterns in time series

Even though discovering patterns in a time series may be lead in four main directions (according to Mueen et al. [1][2][3][3] there are these approaches: Online motif discovery, k-motif discovery, enumeration of motifs of all lengths and fixed length), the concern is to find those subsequences of the time series that repeat themselves throughout it. In their article, Lin et al. [9], proposed the normalization of subsequences in the algorithm to detect the 1-motif. The most suitable is the z-scores, which is defined as:

Definition 9 Z-normalization of a time series T of length n is called a new time series of length n, described as

$$Z = \frac{T - mean(T)}{sd(T)} \quad (9)$$

In time series, when comparing two subsequences, there should be determined a limit to when these subsequences can be considered similar. In their paper, Mueen et al [4] proposed a radius where, if the distance /measure index were in the radius, the two subsequences were to be considered similar. Formally:

Definition 10 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, R a radius and M_i,M_j subsequences of T with length m, M_i is similar to M_j if and only if:

$$d(M_i, M_j) < R \quad (10)$$

, where $R > 0$.

Statistically, is almost 90% probable that a subsequence of length m of a time series has a small

distance (a high similarity) to the subsequence starting from $1,2,\dots,k$ positions after it, for $k < m$. In other words, two adjacent subsequences are presumed to be closely related to each-other.

Definition 11 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, R a radius and M_i, M_j subsequences of T with length m, M_j is a trivial match to M_j if and only if:

$$d(M_i, M_j) < R \text{ and } |i - j| < m \quad (11)$$

Therefore, to find patterns in a time series, it is necessary to eliminate the influences that have adjacent subsequences. Moreover, by eliminating the trivial match $i = j, d(M_i, M_j) > 0$. Therefore, in the algorithm, the radius is applied on the minimal distance between two subsequences.

Definition 12 Given a time series T of length n, $T = (T_1, T_2, \dots, T_n)$, R a radius and M_i, M_j subsequences of T with length m, R' is defined as:

$$R' = \min\{d(M_i, M_j), i \neq j\} + R \quad (12)$$

, where $i, j \in \{1,2,\dots, n - m + 1\}$.

In their article, Dhamo et al.[6] established R to be defined as:

$$R = \frac{sd(M_i)}{\sqrt{m}} = \frac{1}{\sqrt{m}} \quad (13)$$

Therefore, the algorithm for 1-motif detection, is the following:

1-Motif detection algorithm

```
Motif detection_=function(T,m)
    1.#n=length(T)
    2.#R = 1/sqrt(m)
    3. stand={ Ai=(T[i:(i+m-1)]-mean(T[i:(i+m-1)]))/sd(T[i:(i+m-1)]), i = 1:(n-m+1)}
    4.R'=min{d(Ai,Bj)}, Ai=stand[i, 1:m], i=1: (n-m+2) and j=(i+1): (n-m+2)
    5. global_max=0, sim_sub=c(),start_index=0
    6.for i=1: (n-m)
    7.    Ai=stand[i, 1:m]; index=c()
    8.    for j= (i+1) : (n-m+1)
    9.        Bj=stand[j, 1:m]
    10.       if d(Ai, Bj)<R' then inxex=c(index,j)
    11.   end for
    12.   no_adj=find_nr_similar_subseq_without_adj(i,index,m)
    13.   if(reached : global_max) {sim_sub=index,start_index=i}
    14.end for
    15.print start_index, sim_sub
end function
```

, where the function *find_nr_similar_subseq_without_adj)* is described as:

find_nr_similar_subseq_without_adj algorithm

```
find_nr_similar_subseq_without_adj=function(i,index,m)
    1.#n=length(index)
    2.no_adj=c(i); m=1
    3.for j=2:n
    4.    if index[j] - no_adj[k] >m-1, ∀k = 1,2..
    5.        then no_adj=c(no_adj, index[j] )
    6.    end if
    7.end for
    8.return no_adj
end function
```

B. Modification in the algorithm to eliminate the influence of adjacency

1) In the previous algorithm, there are two main problems:

The algorithm can be forced to stop when reached the maximal possible number of detected motifs

In a time series T of length n, the maximal number of subsequences similar to the 1-motif is calculated as:

$$\left\lfloor \frac{(n-m+1)-1}{m-1} \right\rfloor = \left\lfloor \frac{n-m}{m-1} \right\rfloor \quad (14)$$

In other words, the maximal number can be reached if the 1-motif starts at position i=1 and every similar subsequence similar to it ends where another subsequence starts.

For example, in a time series of length n=300 and m=30, the maximal number of subsequences similar to the 1-motif is expected to be:

$$\left\lfloor \frac{300-12}{11} \right\rfloor = \lfloor 26.181 \rfloor = 26 \quad (15)$$

This means that we can use this formula to compare the maximal number of similar subsequences to be discovered to the global maximum. If the maximum is reached, the algorithm stops the search. This enhancement is introduced in the point (15.) of the algorithm 1-motif detection.

2) In the function that excludes adjacent subsequences, the algorithm does not take into consideration a closer proximity of a subsequence starting at the following positions.

Suppose that, during pattern discovery, are found to be similar to it subsequences starting at the following indexes $i + k, k = 0,1,2,\dots,m-2$. The algorithm proposes to remove from the set all the similar subsequences with starting index $i + k, k = 1,2,\dots,m-2$ and to keep as the most similar the i-th subsequence which (not surely) might be less similar than another subsequence belonging to the excluded ones.

What we propose is to make an ordination of all the lengths between the 1-motif and similar subsequences in crescent order. Therefore, the selection will not be made in random, but will be based on proximity in similarity, independently from the position. Therefore, the algorithms will be:

find_nr_similar_subseq_without_adj algorithm

```
find_nr_similar_subseq_without_adj=function(i,index,dist,m)
    1.#n=length(index);
    2.sort(index) #according to their distance
    3.no_adj=c(i); m=1
    4.for j=2:n
    5.    if abs(index[j] - no_adj[k] )>m-1, ∀k = 1,2..
    6.        then no_adj=c(no_adj, index[j] )
    7.    end if
    8.end for
    9.return no_adj
end function
```

The main algorithm, will be transformed into:

1-Motif detection algorithm

Motif detection_=function(T,m)
1.#n=length(T)
2.#$R = \frac{1}{\sqrt{m}}$
3. stand={ $A_i=\frac{T[i:(i+m-1)]-mean(T[i:(i+m-1)])}{sd(T[i:(i+m-1)])}, i = 1:(n-m+1)$}
4.R'=min{d(Ai,Bj)}, A_i=stand[i, 1:m], i=1: (n-m+2) and j=(i+1): (n-m+2)
5. global_max=0, sim_sub=c(),start_index=0
6.for i=1: (n-m)
7. A_i=stand[i, 1:m]; index=c()
8. for j= (i+1) : (n-m+1)
9. B_j=stand[j, 1:m]
10. if d(A_i, B_j)<R' then inxex=c(index,j)
11. end for
12. no_adj=find_nr_similar_subseq_without_adj(i,index,m)
13. local_max=(n-i+1-m)/(m-1)
14. if(reached : global_max) sim_sub=index,start_index=I
15. if(local_max<global_max) then break
16.end for
17. print start_index, sim_sub
end function

In order to make a better comparison between the two algorithms, all the tests were held having the Chouakria index with CID as similarity measure.

IV. COMPARING THE TWO ALGORITHMS

When something changes, it means that there should be made some comparison in order to determine whether the change provided better performance or not. The criteria utilized to compare the two algorithms, are the following: Number of similar subsequences similar to 1-motif, Execution time of the algorithm, Mean distance of similar subsequences to the 1-motif and Mean time to detect a similar subsequence. The tests were made in numerous time series.

A. Number of similar subsequences to the 1-motif

The potential of the algorithm is firstly described by the ability to detect similar subsequences to the 1-motif. The number of similar subsequences to the 1_motif, is described as:

$$no\_sim\_sub = argmax\{d(Ai, Bj) < R', for\ fixed\ i\} \quad (16)$$

, where Ai, Bj are not adjacent.

In 93.55% of the cases, the ability of the algorithms to discover similar subsequences is equal. Graphical results are given in Fig.2.

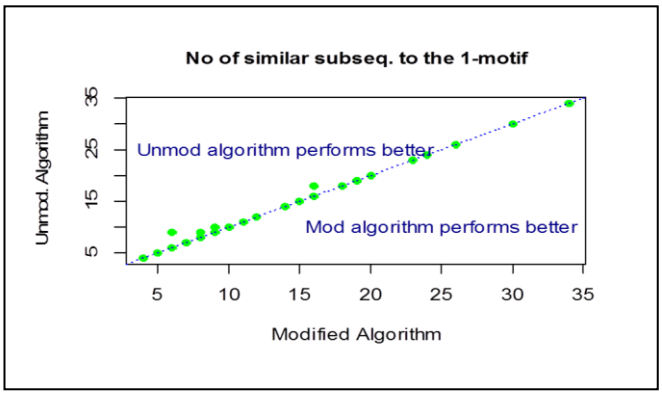

Fig. 2: Comparing the efficiency of the algorithms

Even though in 93.5% of the cases there is no difference in the number of similar subsequences to the 1-motifs found, there is a difference in the set of the selected subsequences in 37.8% of the cases.

TABLE I. DIFFERENCE IN SUBSEQUENCES

No of differences	% different
1	40.9
2	27.3
3	9.1
4	13.6
>4	9.1

It is clear that the potential of the Modified Algorithm does not change from the potential of the Unmodified Algorithm.

B. Comparing the mean distance of the subsequences from the 1-motif

While altering the algorithm, the result will obviously differ. These differences impact the efficiency of the performance. Another important factor in affecting the performance of the algorithm is the mean distance from the 1-motif, which is described as follows:

$$mean\_dist = \frac{1}{l}\sum_{k=1}^{l} d\left(M_i, M_{j[k]}\right) \quad (17)$$

, where M_i is the 1-motif, $M_{j[k]}$ is the k-th subsequence similar to the 1-motif and l the number of similar subsequences to the 1-motif, equal to (16).

In 58.1% of the cases, the mean distance figures out to be the same. In 25.8% of the cases, the mean distance provided by the Unmodified Algorithm is greater than the one provided by the Modified Algorithm (translated as a better performance of the Modified Algorithm) and in the remaining cases (16.1%) the mean distance provided by the

Unmodified Algorithm provides lower ciphers. Graphically, the situation is presented in Fig.3.

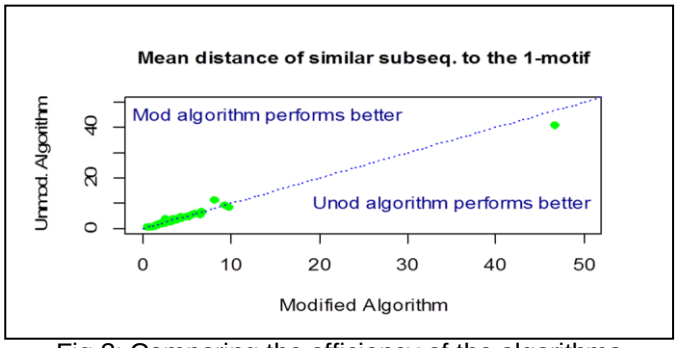

Fig.3: Comparing the efficiency of the algorithms

By dividing the data into two groups: the group where the number of similar subsequences is not the same and where the number of subsequences is the same but changes the subsequences discovered, the situation would be like it follows:

TABLE II. WHERE MEAN DISTANCE IS SMALLER (IN PERCENTAGE)

Result in favor of	No of similar subsequences (%)	
	Different	***The same***
Mod Alg	75	59.1
Unmod Alg	25	40.9

It is clear that in all cases, the Modified Algorithm has shown better performance than the Unmodified Algorithm.

C. Execution time

The most important factor that affects the performance of the algorithm related to its' complexity, is the Execution time, the time required by the system to give a response.

In R, to measure the elapsed time, in the beginning and at the end of the algorithm in R, are added the following commands:

ptm <- proc.time()

proc.time() - ptm

These commands give three parameters:

- user: The necessary time needed to the system to execute user's instruction
- system: The time system requires to call processes
- elapsed : The total time (not necessarily the sum of user time+ system time)

The parameter we used to compare the beneficence of the algorithms is the user time.

In 98.4% of the cases, the execution time was in favor of the Modified Algorithm. In 1.6%, the result was in favor of the Unmodified Algorithm. The comparison is graphically shown in Fig. 4.

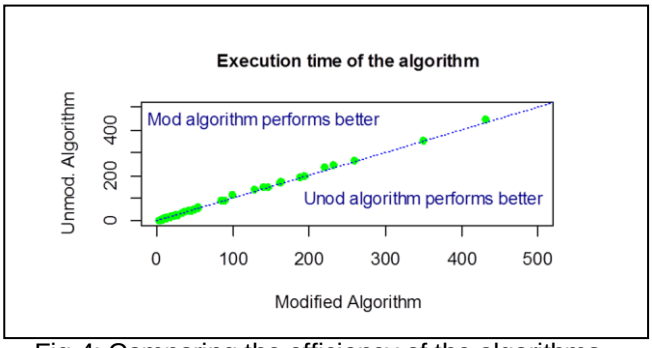

Fig.4: Comparing the efficiency of the algorithms

We compare the mean time to detect a subsequence, determined as below:

$$mean_{time} = \frac{1}{n}T \quad (18)$$

, where T is the elapsed time and n the number of similar subsequences to the 1-motif. The result is 25% in favor of Modified Algorithm, 25% in favor of the Unmodified Algorithm and in 50% of the cases; the mean time to detect a subsequence was equal.

D. Comparing the quality of the similar subsequences detected

Another approach of comparison between the two algorithms is the result itself, the set of subsequences similar to the 1-motif proposed by each algorithm: The Modified and the Unmodified one. An example to how the result differs graphically is given in Fig.5.

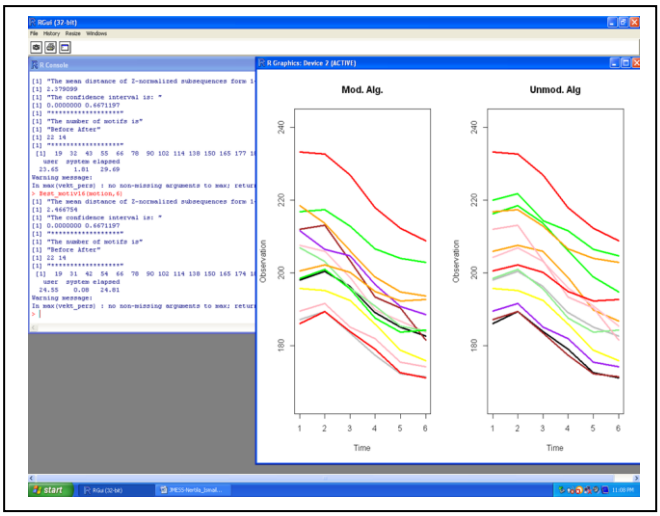

Fig.5: Screenshot to compare the quality of the similar subsequences to the 1-motif

In almost every aspect, the two modifications that were made in the algorithm brought a better performance than the old version in almost every aspect: The execution time was reduced by 1.078 times. Moreover, the ability of the algorithm was almost unchanged. What is more, the modification made possible to reduce the mean distance of the subsequences from the 1-motif by more than 40%.

REFERENCES

The datasets were provided in the following sites:

http://robjhyndman.com/tsdldata/
http://new.censusatschool.org.nz/resource/time-series-data-sets-2013/
https://datamarket.com/data/list/?q=provider%3Atsdl

, package astsa in R, etc.

[1] Mueen A., Keogh, E., J., (2010A): "Online discovery and maintenance of time series motifs". KDD, pg. 1089-1098

[2] Mueen, A., Keogh, E., Zhu, Q., Cash, S., Westover, B., (2009A) "Exact Discovery of Time Series Motifs", SDM, pg. 473-484

[3] Mueen A., Keogh, E., J., Bigdely- Shamlo N., (2009): "Finding Time Series Motifs in Disk-Resident Data". ICDM, pg 367-376

[4] Mueen A., (2013):" Enumeration of Time Series Motifs of All Lengths". ICDM,pg. 547-556

[5] Dhamo, E., Ismailaja, N., Kalluçi, E., (2015): "Comparing the efficiency of CID distance and CORT coefficient for finding similar subsequences in time series", Sixth International Conference ISTI, 5-6 June.

[6] Dhamo, E., Kalluçi, E., Puka, Ll. (2015): "A Complexity Invariant Distance Modification To Discover Similarity In Time Series", Journal of Multidisciplinary Engineering Science and Technology (JMEST) ISSN: 3159-0040 Vol. 2 Issue 6, June - 2015

[7] Batista, G. and wang, X. (2011). "A complexity-invariant distance measure for time series". SIAM Internation Conference on Data Mining (SDM) on Data Mining, Philadelphia, PA, USA.

[8] Chouakria, A., D., Diallo A., Giroud F., (2007) "Adaptive clustering of time series". International Association for Statistical Computing (IASC), Statistics for Data Mining, Learning and Knowledge Extraction, Aveiro, Portugal

[9] Lin, J., Keogh, E. , Lonardi, S. and Patel, P. (2002): "Finding Motifs in Time Series", in Proc. of 2nd Workshop on Temporal Data Mining