Knowledge-Based Systems 21 (2008) 666–671

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Discovering original motifs with different lengths from time series

Heng Tang *, Stephen Shaoyi Liao

Department of Information Systems, City University of Hong Kong, Hong Kong

ARTICLE INFO

Article history:
Received 29 June 2007
Accepted 24 March 2008
Available online 31 March 2008

Keywords:
Time series
Data mining
Pattern discovery
Motif

ABSTRACT

Finding previously unknown patterns in a time series has received much attention in recent years. Of the associated algorithms, the *k*-motif algorithm is one of the most effective and efficient. It is also widely used as a time series preprocessing routine for many other data mining tasks. However, the *k*-motif algorithm depends on the predefine of the parameter *w*, which is the length of the pattern. This paper introduces a novel *k*-motif-based algorithm that can solve the existing problem and, moreover, provide a way to generate the original patterns by summarizing the discovered motifs.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Given the wide utilization of information technology, large amounts of data are being collected during scientific experiments and normal business operations. Data variables collected for many observations at different time periods also result in massive amounts of data. Data from these kinds of observations that are sequentially constructed over time are called a time series. Generally speaking, a time series is a sequence of real numbers where each number represents a value at a given point in time. For example, a sequence could represent a POS (point-of-system) transaction, exchange rates, or weather data over time [1].

Many studies on time series such as seasonality, forecasting, and trend analysis have been carried out from the angle of statistics. In recent years, mining time series has become an important research topic in the data mining and KDD fields [2]. Among these studies, discovering those frequently occurring but previously unknown patterns has received much attention, since it is not only a stand-alone mining process, but also widely used as a data preprocessing routine for other data mining tasks [3]. Some research works use the word "motifs" to refer to the hidden patterns in time series sequences and the proposed algorithms such as the *k*-motif, which has proved to be effective and efficient [3,4].

However, the *k*-motif algorithm is sensitive to the parameter *w*, which is the length of the pattern to be discovered. When the pattern is unknown, the value of *w* is very difficult to estimate. Based on this traditional motif-discovery algorithm, this paper proposes a novel algorithm to solve the above-mentioned problem. Our approach does not require an exact *w* value to be determined in advance, and, moreover, it can be used to identify motifs with different lengths by running it only once. Its effectiveness has been demonstrated by the experimental results in this paper.

The rest of the paper is organized as follows: Section 2 gives a review of the related research studies and their motivations. Section 3 introduces the related definitions as well as the proposed algorithm. Section 4 shows the result of the experiments. Finally, the summary and implications are given in Section 5.

2. Related research and motivations

Discovering patterns from a time series sequence is an important data mining task and much research attention has been devoted to this area [5–9]. Most existing work concentrates on the similarity problem, i.e., it is based on a specific sequence (the keyword sequence) and the attempt to locate similar sequences from the database or similar subsequences for a given sequence.

In some cases, we may also need to identify previously unknown patterns that occur frequently in a time series. The clustering approach has been considered as a natural way to identify the hidden pattern without prior knowledge. Unfortunately, clustering the subsequences of a whole time series sequence proves "meaningless" because of the existence of "trivial matching [10]." However, the research in [3,4] provides an effective alternative called the *k*-motif algorithm. In this algorithm, time series are symbolized and the "random projection" algorithm is used to improve its efficiency. The "trivial match" problem is also solved by this algorithm.

The motif mining algorithm has received considerable attention [11]. It can be regarded as a time series preprocessing procedure and has wide utilization in many other data mining tasks [4]. It provides a solution to pattern discovery in a time series, which is the prerequisite of discovering association rules in a time series

* Corresponding author.
E-mail address: tang.heng@student.cityu.edu.hk (H. Tang).

0950-7051/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2008.03.022

[12]. It can also help the *k*-mean clustering algorithm seed the initial points, as well as facilitate the time series classification algorithms to construct the "typical prototype" [13].

Although the motif-discovery algorithm is one of the most promising approaches in the research area of pattern discovery in time series, it does have some limitations, in particular, the fact that the length of the motif (w) must be determined, or at least estimated, in advance. Since the pattern is previously unknown, this is actually a very difficult task. Furthermore, an inappropriate w value may significantly affect the result of the *k*-motif algorithm, which is illustrated below:

(1) If the value of w is too small (in terms of the length of the hidden pattern), the algorithm can only identify the subsequences of the hidden pattern. The matching process is shown in Fig. 1. When searching and matching the subsequence along the whole sequence, the algorithm identifies a few neighboring subsequences of the pattern (with length w), but fails to reveal the whole pattern.
(2) If the value of parameter w is too large, no pattern can be discovered, since at each round of matching, the excrescent "tail" of the current subsequence can hardly match successfully with the original sequence.
(3) When there are several patterns with different lengths in the time series, the matching task becomes even more difficult. We may need to run the algorithm for each pattern separately, even if a proper w value is provided every time.

In this paper, a novel algorithm is proposed based on the *k*-motif algorithm. Our strategy is to set a relatively small w value to identify the short patterns first, then use a concatenation routine to "concatenate" the discovered short patterns to generate the whole pattern.

One of the drawbacks of the *k*-motif algorithm is that it only identifies each pair of matching patterns but fails to summarize all occurrences of a pattern. In this paper, we introduce an approach to summarize an average pattern based on the occurrences of the pattern.

3. Definitions and algorithm

In this section, we review relevant definitions and propose a novel algorithm for finding motifs with different lengths in time series. Definitions 1–3 are based on the existing work, while the motif-concatenation algorithm and Definitions 4, 5 are given by the authors.

Definition 1 (*Time series*). A time series $T = <t_1, \ldots, t_m>$ is a finite sequence of real-valued variables, where m is the length of the time series.

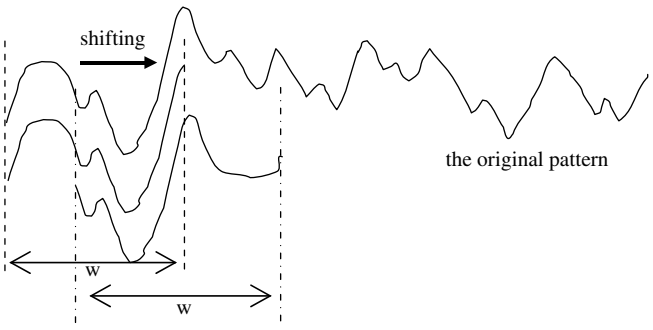

Fig. 1. Searching motif, when w is set smaller than the length of the pattern.

Suppose for any $1 \leqslant p \leqslant q$,$s_p$ is a subsequence of the time series T with length w, i.e., $|s_p| = w$. Then $S = \{s_1, s_2, \ldots, s_q\}$ is the set of all subsequences of T whose collision degrees are going to be considered. It can be noted that not all subsequences with length w of T are included by S since some of them are actually trivial matches and are eliminated by the *k*-motif algorithm.

For any $1 \leqslant i < j \leqslant q$, $s_i, s_j \in S$ are the subsequences of T, supposed $s_i = <s_i, \ldots, s_{i+w-1}>$, $s_j = <s_j, \ldots, s_{j+w-1}>$, respectively (since each subsequence is consecutive and has equal length, the subsequence can be named by its first element without confusion). Without the loss of generality we can suppose $s_i < s_j$, i.e., subsequences in S are ordered by the occurring time of its first element.

Definition 2 (*Collision matrix*). The collision matrix CM of T is a $q \times q$ matrix, and for each element of the matrix denoted as $e_{ij} \in M$, $e_{ij} = collision\_hit(s_i, s_j)$ where *collision_hit* is the similarity degree of two subsequences, for example, s_i and s_j. (The collision_hit algorithm employs a so-called "random projection" method to achieve its efficiency [3,4], but we do not elaborate on that fact in this paper.) In other words, the collision matrix records the similarity degree between any two subsequences $s_i, s_j \in S$, $1 \leqslant i, j \leqslant q$.

The collision matrix is constructed by performing a quite efficient algorithm, the random projection algorithm [4]. The cells in M with the k highest collision degree are regarded as the top-k matches, and the two comparing subsequences of a match are thus called a *motif*. The other cells whose collision degree is smaller than top-k are set to 0. The collision matrix is illustrated in Table 1. Because of its symmetry, the diagonal and "left-down" parts are omitted.

Definition 3 (*Motif*). For any two subsequences s_i, $s_j \in S$ with a length w, $1 \leqslant i, j \leqslant q$, if the collision degree $e_{ij} = collision\_hit(s_i, s_j)$ is among the top-k highest collision degrees of all the matches, we say the pair $M = (s_i, s_j)$ is a *k*-motif of T.

The product of the above-mentioned *k*-motif discovery algorithm records the similarity degree of two subsequences. However, weaknesses of the *k*-motif algorithm still exist.

In particular, the length of the subsequence w must be assigned in advance, which results in three implications: First, if w is too small, the fully matched subsequences of the motif cannot be revealed. Second, if w is set longer than the real length, the tail (unmatched part) of two subsequences will affect the collision hit, which will falsely lead to an "unmatched" result. Third, even if the w is properly specified, only those motifs with length w can be discovered. Therefore, the algorithm fails to provide the correct result if motifs with different lengths are involved in the time series. Fig. 2 gives an illustration of this problem: the two axes represent the subscripts of the two matched subsequences, respectively, and dots in the plot are the cells with top-k collision degrees. However, since the value of w is smaller than it should be, only a section

Table 1
The collision matrix

…	5	0	0	
s3	0	7		
s2	6			
s1				
	s1	s2	s3	…

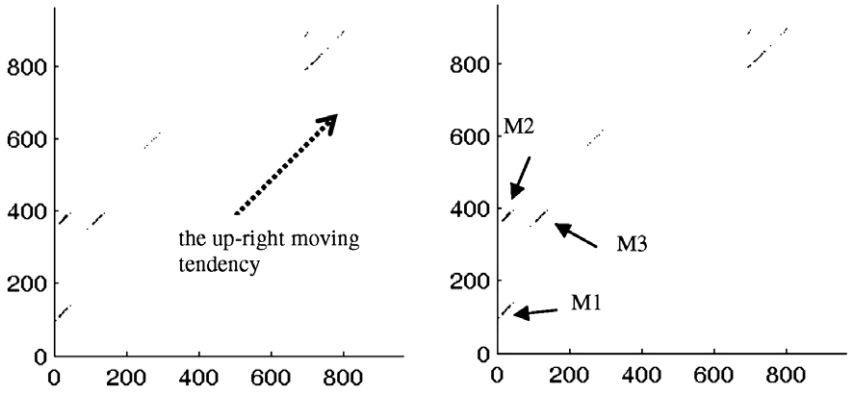

Fig. 2. The collision matrix plot.

of the subsequence is identified. Moreover, the algorithm correctly identifies the matching pairs of the subsections of the subsequences, which results in a series of dots with a tendency to the upper right. As a matter of fact, the shorter w is than it should be, the longer the "dot line" is.

Another weakness of the k-motif algorithm is that, if a hidden pattern occurs for many times in a time series, the algorithm can only regard each two occurrences of the pattern as one motif, and will fail to generate the real pattern of those motifs. For example, $M_1 = (s_i, s_j)$, $M_2 = (s_i, s_k)$, and $M_3 = (s_i, s_j)$ are all k-motifs, but s_i, s_j and s_k in fact originated from the same pattern, i.e., they are three different instances of a pattern. Discovering the pattern can be more interesting and useful than just discovering the motifs. Let's look at the right portion of Fig. 2. Suppose three subsequences, s_1, s_2, and s_3, begin from around 0, 100, and 400, respectively, then we have $M_1 = (s_1, s_2)$, $M_2 = (s_1, s_3)$, and $M_3 = (s_2, s_3)$. However, s_1, s_2, and s_3 actually originated from the same pattern, which is much more interesting than M_1, M_2, and M_3. Suppose a pattern occurs for n times in the original sequence. Normally the k-motif algorithm will identify $\binom{n}{2}/2$ pairs of matching subsequences (there is a "2" in the denominator since the similarity is symmetric).

Motivated by the above problems, the novel method we propose includes three major steps:

Choose a relatively small w value and discover the motifs with length w by executing the original k-motif algorithm.

Concatenate those neighboring motifs that are close enough by lining them up using a line segment.

Identify all motifs that originated from the same pattern, align them, and output the discovered patterns.

In the proposed algorithm, using what we call motif concatenation, the neighboring motifs that are improperly created by too small a w value are concatenated into one segment. The algorithm begins with the most bottom-left motif and searches for the next candidate point in the top-right orientation.

Algorithm. Motif concatenation

Input: Time series T, the set of all subsequences S, collision matrix CM. The search area width d and slope range $[\alpha 1, \alpha 2]$ is used to constrain the area of search as illustrated in Fig. 2;
setM = {Mi|$M_i = (s_{i1}, s_{i2})$ is a motif of T} is the set of all identified motifs, and setM elements are multiple sorted by the motifs' two subsequences.
Output: A partition of setM, say, $setM = \text{seg}_1 \cup \text{seg}_2 \cup \ldots \cup \text{seg}_q$, and each segment $\text{seg}_i = \langle M_{i1}, M_{i2}, \ldots, M_{ip} \rangle$ is an ordered set of neighboring motifs.
BEGIN
For any motif $M_i = (s_{i1}, s_{i2})$, add a new attribute "segid" to all motifs with initial value 0, which is used to indicate to which segment they belong.
For each $M_i \in setM$, $M_i = (s_{i1}, s_{i2})$ Do
  If M_i.segid = 0 then M_i.segid = new(segid)
//If the segid=0, then assign a new segid, which means a new segment starts.
  For each $M_j \in setM, j > i$, if $|s_{i1} - s_{j1}| \leqslant d$ and $|s_{i2} - s_{j2}| \leqslant d$ then
    //If is within the "search square"
    slope $= (s_{j2} - s_{j1})/(s_{i2} - s_{i1})$
    If slope $\in [\alpha 1, \alpha 2]$ then
    //If the slope is between the two thresholds
      If M_j.segid = 0 then M_j. segid = M_i.segid
      //If it hasn't been assigned to any other segments, assign the segid of M_i
  End for
End for
For each segment ID segid_i Do
  Put all motifs that have the segid_i into seg_i
End for
END.

The idea of the algorithm is straightforward and is pictured in Fig. 3: assuming the current motif is the origin, the area of searching for the next neighboring motif for concatenation is represented by the shadow area.

The worst time complexity of this algorithm is $O((|\text{setM}|^2 - |\text{setM}|)/2)$, even if the constraining shadow area is not considered. The time cost of the algorithm can virtually be ignored since |setM| is generally quite small. By executing the algorithm, all motif points that are close enough and show a diagonal tendency will be assigned the same segid, indicating the furthest extensions of the segment. This algorithm actually merges those neighboring motifs into a longer one, as illustrated in Fig. 4.

Once all motif segments are obtained, the next task will be identifying those segments that originated from the same hidden pattern.

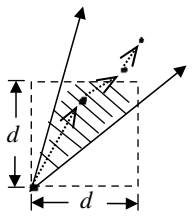

Fig. 3. Searching and concatenating motifs.

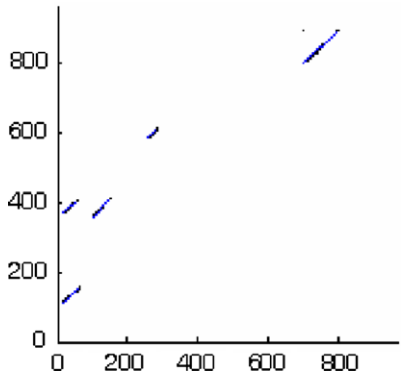

Fig. 4. Concatenate neighboring motifs to segment.

The idea of summarizing motif segments is as follows: if segi and segj have an overlap on the axis *y*, we may conclude that very likely they originated from the same pattern, since there must exist $(s_i,t_i) \in \text{seg}_i$ and $(s_j,t_j) \in \text{seg}_j$ such that $t_i = t_j$. Because s_i is similar to t_i and s_j is similar to t_j, then s_i and s_j are very likely to be similar. Therefore, the three subsequences s_i, s_j, and t_j need to be summarized to find the pattern from which they originated. It can be noted that the similarity relationships among those motif segments are not mathematically transitive due to the native characteristics of the "random projection" algorithm. Therefore, only an empirical conclusion is given.

Definition 4 (*Segments overlap*). For segments $\text{seg}_i = \langle M_{i1}, M_{2p}, \ldots, M_{ip}\rangle = \langle (s_{i1}, t_{i1}), (s_{i2}, t_{i2}), \ldots, (s_{ip}, t_{ip})\rangle$ and segments $\text{seg}_j = \langle (s_{j1}, t_{j1}), (s_{j2}, t_{j2}), \ldots, (s_{jq}, t_{jq})\rangle$, the overlap of seg_i and seg_j is characterized by:

$$x\_\text{overlap}(\text{seg}_i, \text{seg}_j) = 2 \times (\min(s_{jp}, s_{ip}) - \max(s_{j1}, s_{i1}))/(s_{jp} + s_{ip} - s_{i1} - s_{j1})$$

$$y\_\text{overlap}(\text{seg}_i, \text{seg}_j) = 2 \times (\min(t_{jp}, t_{ip}) - \max(t_{j1}, t_{i1}))/(t_{jp} + t_{ip} - t_{i1} - t_{j1})$$

These two measures are used to characterize the overlapping degree along the *x*-axis and *y*-axis, respectively.

Since the collision matrix is symmetric with respect to the main diagonal, and only its top-left half is stored, we may need to examine both x_overlap and y_overlap degrees to see whether two segments are actually overlapping. For example, seg_i and seg_j in Fig. 5 are segments and seg_i' and seg_j' are their "image segments" with respect to the main diagonal, y_overlap (seg_i, seg_j) cannot be observed because the bottom-right part is omitted, but we can still observe x_overlap (seg_i', seg_j').

Definition 5 (*Equivalent class of segments*). Suppose SEG is a set of segments, if for any two segments seg_i, $\text{seg}_j \in \text{SEG}, i \neq j$, $x\_\text{overlap}(\text{seg}_i, \text{seg}_j) \leqslant \theta$ or $y\_\text{overlap}(\text{seg}_i, \text{seg}_j) \leqslant \theta$ is satisfied, then SEG is said to be an equivalent class of segments.

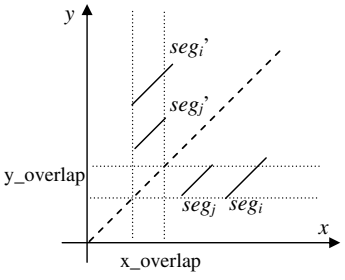

Fig. 5. Symmetry of motif segments.

By examining the x_overlap degree and y_overlap degree, we can easily partition the set of all discovered segments into equivalent classes, and each class represents the pattern of its member segments.

Once the classes are obtained, we align each segment in each class by cropping the matched part of the segments; the falsely included parts at the beginning or end of each segment need to be cut off and only the common part is kept. The pattern from which the segments originated is obtained by calculating the average of all subsequences in a class. This idea is shown in Fig. 6.

4. Experimental results

4.1. Experiment A

This experiment consists of two stages. In the first stage, we run the original algorithm on the testing dataset and results are explained. In the second stage, the proposed algorithm is applied to the same dataset to illustrate its advantages. The basic testing dataset is created by the *cylinder–bell–funnel* (c–b–f) synthetic approach, which is widely used as the testing dataset for time series analysis [14]. The three different shapes (cylinder, bell, and funnel) and an additional synthetic shape "m" are randomly picked up and inserted into a sequence of noise. The data sequence is illustrated in Fig. 7, and we give each inserted subsequence an ID for facilitating narration.

In the dataset, the length of each shape (pattern) "c", "b," and "f" is 64 whereas the length of "m" is 100. First, we run the *k*-motif algorithm proposed in [3,4] on this dataset three times. Each time, the parameter *w*, which is the length of subsequences, is set to 64, 40, and 105, respectively.

When $w = 64$, the algorithm perfectly identifies the motifs (f1, f2), (c1, c2), (c1, c3), (c2, c3), (b1, b2). The motif (m1, m2) as a whole is not successfully identified. As a result, a series of points can be observed in the plot of the collision matrix. Each of those points represents a subsequence of the original pattern.

When $w = 40$, none of the motifs as a whole is identified. Some intervals with 45° of slope can be found in the collision matrix plot and each of them represents a subsequence of the original pattern. This implies that only the header parts of each motif are discovered.

When $w = 105$, the algorithm identifies motif (m1, m2) but fails to find the other motifs whose matching subsequences are shorter than *w*.

In the second stage, the proposed algorithm is applied to the same dataset. The same different values of *w* as 64, 40, and 105 are used. Parameter *d* is set to $1.25^* w$ and $\alpha_1 = 0.65$, $\alpha_2 = 1.35$.

When $w = 64$, the motif algorithm successfully identifies the motifs (f1, f2), (c1, c2), (c1, c3), (c2, c3), (b1, b2). Furthermore, c1, c2, and c3 are identified to have originated from the same pattern (Fig. 8). The motif (m1, m2) is also discovered by using the concatenation approach.

When $w = 40$, we get similar results as $w = 64$. Based on the concatenation algorithm, the whole matching subsequences are identified. The pattern of c1, c2, c3 is also discovered.

When $w = 105$, only motif (m1, m2) is identified.

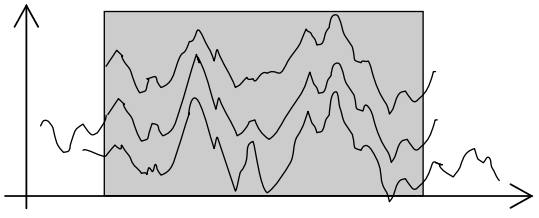

Fig. 6. Cropping segments.

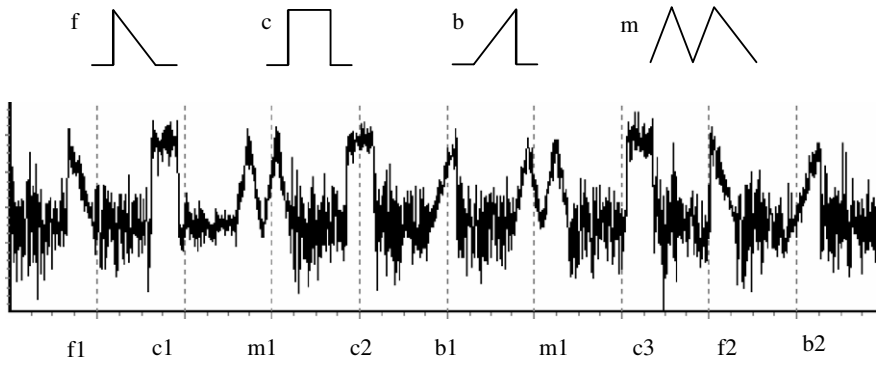

Fig. 7. The synthetic c-b-f-m dataset.

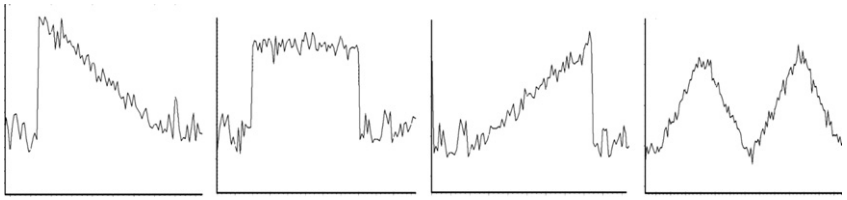

Fig. 8. Identified patterns (f, c, b, m).

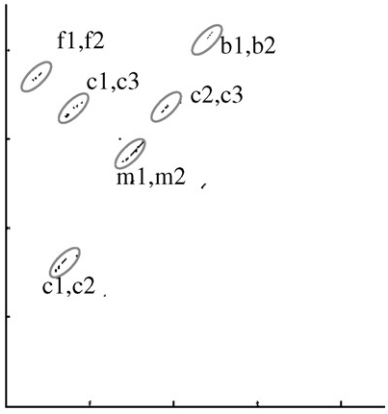

Fig. 9. The plot of the collision matrix when $w = 40$.

The plot of the collision matrix is shown in Fig. 9, where the value of w is smaller than the length of the matching sequences should have, a series of dots with the direction along the main diagonal is shown. Nevertheless, the proposed algorithm still identifies all full hidden patterns, even when a relatively small w value is given.

4.2. Experiment B

In this experiment, a real dataset "blowfly.dat" is used, which is publicly available at the time series data library [15]. We run the proposed algorithm three times with w value 20, 30, and 50 where parameter d is set to $1.25^{*} w$ and $\alpha_1 = 0.65$, $\alpha_2 = 1.35$. The final result proves that when the value of w is 20 or 30, one motif is identified as shown in Fig. 10. Although the length of the matched subsequences in the motif is about 37, the motif is still successfully identified.

It is worth noting that the three parameters (d, α_1, and α_2) constrain the searching area in the concatenation algorithm and have a significant impact on the result. A large searching area facilitates identification of the next matching point but may also falsely include noise points. Therefore, in order to improve the discovery ability, the three parameters d, α_1, and α_2 need to be adjusted accordingly. Nevertheless, adjusting these parameters, which are not quite sensitive with regard to the algorithm, is still much easier than specifying a proper value to w because the exact lengths of the hidden patterns are generally unknown in advance. In these two experiments we even adopt the same group of parameter values, which implies the insensitivity of the parameter settings.

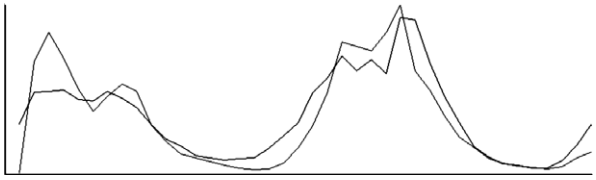

Fig. 10. Matched subsequences of the identified motifs.

5. Conclusion

The focus of this paper is placed on a novel motif-discovery algorithm. The major contribution of our work is the proposition of a novel approach to improve the widely used k-motif algorithm, which suffers from the problem of the setting of parameter w. More importantly, the conventional k-motif approach can only discover patterns with a predefined length, which is normally only a fraction of the original patterns. In contrast, the proposed approach is capable of discovering the original whole patterns with different lengths. Experimental results prove the effectiveness of the approach. As for future research, we plan to extend the proposed approach to deal with the rule-discovery problem in time series.

Acknowledgements

We thank Dr. Eamonn Keogh who provided the source code of the original motif-discovery algorithm. This research is directly supported by a CERG grant (CityU1236/03E) of RGC, Hong Kong SAR.

References

[1] S.B. L.M. Liu, S.L. Sclove, R. Chen, W.J. Lattyak, Data mining on time series: an illustration using fast-food restaurant franchise data, Computational Statistics & Data Analysis 37 (2001).

[2] Y. K. M. Last, and A. Kandel, Knowledge discovery in time series databases, IEEE Trans on System, Man and Cybernetics, pp. 160-169, 2001.
[3] K. Lin J., E., Patel, P. and Lonardi, S., Finding Motifs in Time Series, presented at In the 2nd Workshop on Temporal Data Mining, at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002.
[4] B. Chiu, E. Keogh, and S. Lonardi, Probabilistic Discovery of Time Series Motifs, presented at In Proceedings of the 9th International Conference on Knowledge Discovery and Data mining, Washington, D.C., 2003.
[5] X. Ge and P. Smyth, Deformable Markov model templates for time-series pattern matching, presented at proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, 2000.
[6] E. Keogh, K. Chakrabarti, M. Pazzani, S. Mehrotra, Dimensionality reduction for fast similarity search in large time series databases, Journal of Knowledge and Information Systems (2000).
[7] S. S. Liao, H. Tang, and W.-Y. Liu, Finding Relevant Sequences in Time Series Containing Crisp, Interval and Fuzzy Interval Data, IEEE Transactions on System, Man and Cybernetics (B), vol. 34, 2004.
[8] H. Mannila, H. Toivonen, and A. I. Verkamo, Discovering frequent episodes in sequences, presented at Proc. 1st Int. Conf. Knowledge Discovery and Data Mining, Montreal, Canada, 1995.
[9] M. Vlachos, G. Kollios, and D. Gunopulos, Discovering similar multidimensional trajectories, presented at proceedings 18th International Conference on Data Engineering, 2002.
[10] E. Keogh, Lin, J. and Truppel, W., Clustering of Time Series Subsequences is Meaningless: Implications for Past and Future Research, presented at proceedings of the 3rd IEEE International Conference on Data Mining, Melbourne, Nov 19-22, 2003.
[11] E. Keogh. http://www.cs.ucr.edu/~eamonn/selected_publications.htm.
[12] K.-I. L. G. Das, H. Mannila, G. Renganathan, and P. Smyth, Rule discovery from time series, presented at The 4th International Conference of Knowledge Discovery and Data Mining, 1998.
[13] E. Keogh and M. Pazzani, An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback, presented at proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, New York, 1998.
[14] W. Kadous. http://www.cse.unsw.edu.au/~waleed/phd/html/node119.html.
[15] R.J. Hyndman. http://www-personal.buseco.monash.edu.au/~hyndman/TSDL/.