Optimized Multi-resolution Indexing and Retrieval Scheme of Time Series

Muhammad Marwan Muhammad Fuad(✉)

Forskningsparken 3, Institutt for Kjemi, NorStruct,
The University of Tromsø – The Arctic University of Norway, 9037 Tromsø, Norway
marwan.fuad@uit.no

Abstract. Multi-resolution representation has been successfully used for indexing and retrieval of time series. In a previous work we presented Tight-MIR, a multi-resolution representation method which speeds up the similarity search by using distances pre-computed at indexing time. At query time Tight-MIR applies two pruning conditions to filter out non-qualifying time series. Tight-MIR has the disadvantage of storing all the distances corresponding to all resolution levels, even those whose pruning power is low. At query time Tight-MIR also processes all stored resolution levels. In this paper we optimize the Tight-MIR algorithm by enabling it to store and process only the resolution levels with the maximum pruning power. The experiments we conducted on the new optimized version show that it does not only require less storage space, but it is also faster than the original algorithm.

Keywords: Multi-resolution indexing and retrieval · Optimization · Tight-MIR · Time series

1 Introduction

A time series is a chronological collection of observations. The particular nature of these data makes them more appropriate to be handled as whole entities rather than separate numeric observations. In the last decade a great deal of research was devoted to the development of time series data mining because of its various applications in finance, medicine, engineering, and other domains.

Time series are usually represented by *Dimensionality Reduction Techniques* which map the time series onto low-dimension spaces where the query is processed.

Several dimensionality reduction techniques exist in the literature, of those we mention: *Piecewise Linear Approximation* (PLA) [1], and *Adaptive Piecewise Constant Approximation* (APCA) [2].

Multi-resolution dimensionality reduction techniques map the time series to several spaces instead of one. In a previous work [3] we presented a multi-resolution indexing and retrieval method of time series called Weak-MIR. Weak-MIR uses pre-computed distances and two filters to speed up the similarity search. In [4] we presented another multi-resolution indexing and retrieval method, MIR-X, which associates our multi-resolution approach with another dimensionality reduction technique. In a third work

DOI: 10.1007/978-3-319-23485-4_61

[5] we introduced Tight-MIR which has the advantages of the two previously mentioned methods. Tight-MIR, however, stores distances corresponding to all resolution levels, even though some of them might have a low pruning power. In this paper we present an optimized version of Tight-MIR which stores and processes only the resolution levels with the maximum pruning power.

The rest if the paper is organized as follows: Section 2 is a background section. The optimized version is presented in Section 3 and tested in Section 4. We conclude this paper with Section 5.

2 Background

In [3] we presented *Multi-resolution Indexing and Retrieval Algorithm* (Weak-MIR). The motivation behind this method is that traditional dimensionality reduction techniques use a "one-resolution" approach to indexing and retrieval, where the dimension of the low-dimension space is selected at indexing time, so the performance of the algorithm at query time depends completely on the choice made at indexing time. But in practice, we do not necessarily know a priori the optimal dimension of the low-dimension space.

Weak-MIR uses a multi-resolution representation of time series. During indexing time the algorithm computes and stores distances corresponding to a number of resolution levels, with lower resolution levels having lower dimensions. The algorithm uses these pre-computed distances to speed up the retrieval process. The basis of Weak-MIR is as follows: let U be the original n-dimension space and R be a $2m$-dimension space, where $2m \leq n$. At indexing time each time series $u \in U$ is divided into m segments each of which is approximated by a function (we used a first degree polynomial in [3]) so that the approximation error between each segment and the corresponding polynomial is minimal. The n-dimension vector whose components are the images of all the points of all the segments of a time series on that approximating function is called the *image vector* and denoted by $\bar{u}$. The images of the two end points of the segment are called the *main image* of that segment. The $2m$ main images of each time series are the *projection vector* u^R.

Weak-MIR uses two distances, the first is d which is defined on a n-dimension space, so it is the distance between two time series in the original space, i.e. $d(u_i, u_j)$, or the distance between the original time series and its image vector, i.e. $d(u_i, \bar{u}_i)$. The second distance is d^R which is defined on a $2m$-dimension space, so it is the distance between two projection vectors, i.e. $d^R(u_i^R, u_j^R)$. We proved in [3] that d^R is a lower bound of d when the Minkowski distance is used.

The resolution level k is an integer related to the dimensionality of the reduced space R. So the above definitions of the projection vector and the image vector can be extended to further segmentation of the time series using different values $\leq m_k$. The image vector and the projection vector at level k are denoted by $\bar{u}^{(k)}$ and $u^{R(k)}$, respectively.

Given a query(q, ε), let $\bar{u}, \bar{q}$ be the projection vectors of u, q, respectively, on their approximating functions, where $u \in U$. By applying the triangle inequality we get:

$$|d(u, \bar{u}) - d(q, \bar{q})| > \varepsilon \tag{1}$$

This relation represents a pruning condition which is the first filter of Weak-MIR.

By applying the triangle inequality again we get:

$$d^R\left(u^{R(k)}, q^{R(k)}\right) > \varepsilon + d\left(q, \bar{q}^{(k)}\right) + d\left(u, \bar{u}^{(k)}\right) \tag{2}$$

This relation is the second filter of Weak-MIR.

In [4] we introduced MIR-X which combines a representation method with a multi-resolution time series. MIR-X uses one of the two filters that Weak-MIR uses together with the low-dimension distance of a time series dimensionality reduction technique. We showed how MIR-X can boost the performance of Weak-MIR.

In [5] we presented Tight-MIR which has the advantages of both Weak-MIR and MIR-X in that it is a standalone method, like Weak-MIR, yet it has the same competitive performance of MIR-X. In Tight-MIR instead of using the projection vector to construct the second filter, we access the raw data in the original space directly using a number of points that corresponds to the dimensionality of the reduced space at that resolution level. In other words, we use $2m$ raw points, instead of $2m$ main images, to compute d^R. There are several advantages to this modification; the first is that the new d^R is obviously tighter than d^Ras computed in [4]. The second is that when using a Minkowski distance d^R is a lower bound of the original distance in the original space. The direct consequence of this is that the two distances $d\left(q, \bar{q}^{(k)}\right), d\left(u, \bar{u}^{(k)}\right)$ become redundant, so the second filter is overwritten by the usual lower bounding condition $d^R\left(u^{R(k)}, q^{R(k)}\right) > r$.

At indexing time the distances $d\left(u, \bar{u}^{(k)}\right)$ $\forall\, u \in U$ are computed and stored. At query time the algorithm starts at the lowest level and applies (1) to the first time series in U. If the time series is filtered out the algorithm moves to the next time series, if not, the algorithm applies equation (2). If all the time series in the database have been pruned the algorithm terminates, if not, the algorithm moves to a higher level.

Finally, after all levels have been exploited, we get a candidate answer set which we then scan sequentially to filter out all the non-qualifying time series and return the final answer set.

3 An Optimized Multi-resolution Indexing and Retrieval Scheme

The disadvantage of the indexing scheme presented in the previous section is that it is "deterministic", meaning that at indexing time the time series are indexed using a top-down approach, and the algorithm behaves in a like manner at query time. If some resolution levels have low utility in terms of pruning power, the algorithm will still use the pre-computed distances related to these levels, and at query time these levels will also be examined. Whereas the use of the first filter does not require any query time distance evaluation, applying the second does include calculating distances and thus we might be storing and calculating distances for little pruning benefit.

We propose in this paper an optimized multi-resolution indexing and retrieval scheme. Taking into account that the time series to which we apply equations (1) and (2) are those which have not been filtered out at lower resolution levels, this optimized scheme should determine the optimal combination of resolution levels the algorithm should keep at indexing time and consequently use at query time.

The optimization algorithm we use to solve this problem is the *Genetic Algorithm*. The Genetic Algorithm (GA) is a famous evolutionary algorithm that has been applied to solve a variety of optimization problems. GA is a population-based global optimization algorithm which mimics the rules of Darwinian selection in that weaker individuals have less chance of surviving the evolution process than stronger ones. GA captures this concept by adopting a mechanism that preserves the "good" features during the optimization process.

In GA a population of candidate solutions (*chromosomes*) explores the search space and exploits this by sharing information. These chromosomes evolve using genetic operations (selection, crossover, mutation, and replacement).

GA starts by randomly initializing a population of chromosomes inside the search space. The fitness function of these chromosomes is evaluated. According to the values of the fitness function new offspring chromosomes are generated through the aforementioned genetic operations. The above steps repeat for a number of generations or until a predefined stopping condition terminates the GA.

The new algorithm, which we call *Optimized Multi-Resolution Indexing and Retrieval* – O-MIR, works as follow; we proceed in the same manner described for Tight-MIR to produce k candidate resolution levels. The next step is handled by the optimizer to select op resolution levels of the k resolution levels, where these op levels provide the maximum pruning power. For the current version of our algorithm the number of resolution levels to be kept, op, is chosen by the user according to the storage and processing capacity of the system. In other words, our algorithm will decide which are the op optimal resolution levels to be kept out of the k resolution levels produced by the indexing step.

Notice that when $op = 1$ we have one resolution level, which is the case with traditional dimensionality reduction techniques.

The optimization stage of O-MIR starts by randomly initializing a population of chromosomes $V_j = \langle v_1^j, v_2^j, \dots, v_{op}^j \rangle$ where $j = 1, \dots, popSize$ and where $v_1^j < v_2^j < \cdots < v_{op}^j$. Each chromosome represents a possible configuration of the resolution levels to be kept. The fitness function of our optimization problem is the pruning power of this configuration. As in [5], the performance criterion is based on the latency time concept presented in [6]. The latency time is calculated by the number of cycles the processor takes to perform the different arithmetic operations (>,+ - ,*,abs, sqrt) which are required to execute the similarity search query. This number for each operation is multiplied by the latency time of that operation to get the total latency time of the similarity search query. The latency time is 5 cycles for (>, + -), 1 cycle for (abs), 24 cycles for (*), and 209 cycles for (sqrt) [6]. The latency time for each chromosome is the average of the latency time of nQ random queries.

As with other GAs, our algorithm selects a percentage $sRate$ of chromosomes for mating and mutates a percentage $mRate$ of genes. The above steps repeat for $nGen$ generations.

4 Experiments

We compared O-MIR with Tight-MIR on similarity search experiments on different time series datasets from different time series repositories [7], and [18] using different threshold values, and for different values of op. Since the value of op is related to the value of k, which in turn depends on the length of the time series tested, we denote the percentage of the resolution levels kept to the total resolution levels by $pk = op/k$.

As for the parameters of the algorithm that we used in the experiments; the population size, $popSize$, was 16, the number of generations $nGen$ was 100, the mutation rate, $mRate$, was 0.2, the selection rate, $sRate$, was 0.5, and the number of queries, nQ, was set to 10.

We show in Fig. 1 the results of our experiments. For (CBF), (Wafer), and (GunPoint) we have 5 resolution levels ($k = 5$). For these datasets we chose $pk = 40\%, 60\%, 80\%$. (motoCurrent) has 8 resolution levels ($k = 8$), we chose $pk = 25\%, 50\%, 75\%$ As we can see, the results are promising in terms of latency time and storage space. For the three first datasets O-MIR is faster than Tight-MIR and in addition, it required less storage space. This is also the case with (motoCurrent) except for $pk = 20\%$ where the latency time for O-MIR was longer than that of Tight-MIR. However, for this value of pk the gain of storage space is substantial without much increase in latency time.

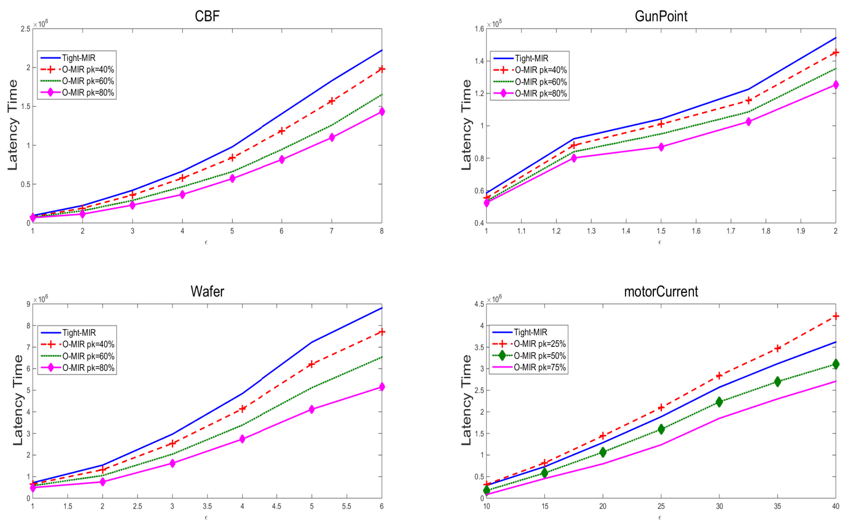

Fig. 1. Comparison of the latency time between Tight-MIR and O-MIR on datasets (CBF), (Wafer), (GunPoint)), and (motoCurrent) for different values of pk

5 Conclusion

In this paper we presented an optimized version of our previous Tight-MIR multi-resolution indexing and retrieval method of time series. Whereas the original method stores and processes all the resolution levels the indexing step produces, the new algorithm, O-MIR, optimizes this process by applying the genetic algorithms to choose the resolution levels with the maximum pruning power. The experiments we conducted show that O-MIR is faster than Tight-MIR, and it also has the advantage of requiring less storage space.

We believe that the main advantage of the new method is that it reduces the storage space requirement of the original method which can be a burden when applying such multi-resolution methods to large datasets.

References

1. Morinaka, Y., Yoshikawa, M., Amagasa, T., Uemura, S.: The L-index: an indexing structure for efficient subsequence matching in time sequence databases. In: Proc. 5th Pacific Asia Conf. on Knowledge Discovery and Data Mining (2001)
2. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrotra, S.: Locally Adaptive Dimensionality Reduction for Similarity Search in Large Time Series Databases. SIGMOD (2001)
3. Muhammad Fuad, M.M., Marteau P.F.: Multi-resolution approach to time series retrieval. In: Fourteenth International Database Engineering & Applications Symposium– IDEAS 2010, Montreal, QC, Canada (2010)
4. Muhammad Fuad, M.M., Marteau P.F.: Speeding-up the similarity search in time series databases by coupling dimensionality reduction techniques with a fast-and-dirty filter. In: Fourth IEEE International Conference on Semantic Computing– ICSC 2010, Carnegie Mellon University, Pittsburgh, PA, USA (2010)
5. Muhammad Fuad, M.M., Marteau, P.F.: Fast retrieval of time series by combining a multi-resolution filter with a representation technique. In: The International Conference on Advanced Data Mining and Applications–ADMA 2010, ChongQing, China, November 21, 2010
6. Schulte, M.J., Lindberg, M. Laxminarain, A.: Performance evaluation of decimal floating-point arithmetic. In: IBM Austin Center for Advanced Studies Conference, February 2005
7. http://povinelli.eece.mu.edu/
8. Keogh, E., Zhu, Q., Hu, B., Hao, Y., Xi, X., Wei, L., Ratanamahatana, C.A.: The UCR Time Series Classification/Clustering (2011). www.cs.ucr.edu/~eamonn/time_series_data/