Available online at www.sciencedirect.com
SCIENCE DIRECT®
European Journal of Operational Research 173 (2006) 921–937
EUROPEAN JOURNAL OF OPERATIONAL RESEARCH
www.elsevier.com/locate/ejor

Mining time series data for segmentation by using Ant Colony Optimization

Sung-Shun Weng *, Yuan-Hung Liu

Department of Information Management, Fu Jen Catholic University, Hsin-Chuang, Taipei 242, Taiwan, ROC

Received 30 October 2004; accepted 2 September 2005
Available online 22 November 2005

Abstract

In trying to distinguish data features within time series data for specific time intervals, time series segmentation technology is often required. This research divides time series data into segments of varying lengths. A time series segmentation algorithm based on the Ant Colony Optimization (ACO) algorithm is proposed to exhibit the changeability of the time series data. In order to verify the effect of the proposed algorithm, we experiment with the Bottom-Up method, which has been reported in available literature to give good results for time series segmentation. Simulation data and genuine stock price data are also used in some of our experiments. The research result shows that time series segmentation run by the ACO algorithm not only automatically identifies the number of segments, but its segmentation cost was lower than that of the time series segmentation using the Bottom-Up method. More importantly, during the ACO algorithm process, the degree of data loss is also less compared to that of the Bottom-Up method.

Keywords: Artificial intelligence; Data mining; Time series; Ant Colony Optimization

1. Introduction

In recent years, technological developments have allowed corporations and organizations to rapidly accumulate massive amounts of data to make information analysis and knowledge extraction from mass information possible. Data mining is necessary for the extraction and analysis of such massive amounts of data, and its application is widely used in fields ranging from medical analysis to weather forecasts, to business, etc.

The data that is gathered for data mining can be generally classified into two different types of data: static and dynamic. Last et al. [19] pointed out the differences between the two: in static databases, an attribute value in a record is independent of the values of the same attribute in other data records. In dynamic databases, some attributes are

* Corresponding author. Tel.: +886 2 29052717; fax: +886 2 29052182.
E-mail address: im1032@mails.fju.edu.tw (S.-S. Weng).

0377-2217/$ - see front matter
doi:10.1016/j.ejor.2005.09.001

meaningful only after records are arranged in a time sequence. The time series data are data in a dynamic database and are generally defined as "a series of data ordered in time sequence but segmented by fixed time intervals" [24]. Such information can be found everywhere in life—in monthly trading data, in the average daily temperature, in a company's stock closing prices, in seasonal sales, in a highway's hourly vehicle passing rate, etc.

By recording and analyzing time series data, an organization or an individual can understand the changes (or behavior) in an environment better, and thus respond with appropriate decisions. On the other hand, the Internet, various databases and other information technologies have made information accumulation quick and convenient, but the enormous volume of data has made the task of finding changing data patterns complicated and inefficient. The difficulty is the same when finding similar-patterned time series in a large database.

Han and Kamber [11] suggest that making use of Euclidean distance can transfer the time domain into a frequency domain in order to calculate time series similarities. In other words, by describing the changing processes of data with frequency expressions (such as Discrete Fourier Transform (DFT) or Discrete Wavelet Transform (DWT) expressions), the distance measurement method can be used for simple and effective calculation in analyzing data similarities. In addition, scholars have studied historical stock prices and have found out that when varying time series data are clustered according to similarity, and if a complete time series is divided into separate segments before clustering, the result becomes significantly better than the non-segmented time series data [8,13].

Time series segmentation classifies similar data to form a segment. Furthermore, time series data can be segmented into non-overlapping data of different time durations which allow each given segment feature to describe original time series data. Thus, segmented series data mining not only reduces storage space, but it also outperforms the results of original data mining.

While a better time series segmentation reduces the amount of data, it also keeps the data's original information. According to the literature review, current time series compression techniques require expert understanding prior to determining segmentation methods, while appropriate threshold values need to be adopted in order to reduce information loss.

The number of segments is the key to the time series segmentation threshold values. Current statistical methods (time series data transformation methods) often require data that have been processed manually. Therefore, even if the best threshold value is gained from the trial-and-error method, such a result is often affected by subjective judgment. Fortunately, thanks to modern technology, artificial intelligence has advanced. In the 1990s, some scholars took ants' behavior as a teaching guide from nature. They developed a set of answer-finding solutions based on how ants find food, and this they called the Ant Colony Optimization (ACO) method [4], which has greatly influenced this research. Coupled with the ACO algorithm, an artificial intelligence approach is used to select threshold values in unsupervised calculations for time series data segmentation to reduce human involvement.

2. Literature review

There are some applications of the Ant Colony Optimization in data mining. Parpinelli et al. [21,22] proposed an algorithm for rule discovery in Ant-Miner databases. The algorithm extracts classification rules from data to be applied as a decision aid to unseen data. Results show that the proposed algorithm is able to achieve good predictive accuracy as well as reduce the number of rules at the same time.

Last et al. [19] developed a time series data mining process using signal processing as well as an Information-Theoretic fuzzy approach. During the process, the time series data mining is divided into three major steps: preprocessing, data mining, and post processing. There are two major processes during data handling: data cleaning and feature extraction, with time series segmentation being a technique. Zhang and Qi [29] investigated the issue of how to effectively model the time series with both seasonal and trend patterns. In particu-

lar, they studied the effectiveness of data preprocessing, neural network modeling and forecasting performance.

Looking at segment lengths after the time series segmentation process, it is evident that the method uses two different types of segment classification. The first type is the equal-length segmentation, which literally means segmenting the time series into identical lengths through distance calculation where similarities among different segments are identified [15]. The other type divides time series into different lengths, which is mainly used for features extraction.

2.1. Equal-length segmentation

Equal-length segmentation is easily done through piecewise aggregate approximation (PAA) [17]. A time series X with length n is divided into N equal portions. The average data value in each segment is used to represent the segment's data feature. Another method, the Discrete Haar Wavelet Transform (DWT) calculation [17], continually divides time series into two equal portions. Every time a division process is performed, the number of segments doubles. The length of segments is then shortened to half of the previous ones. During segmentation, deviations from the average values are recorded for every bi-segmentation in each level. The recorded values are then adopted as coefficients in a new series. Therefore, at the ith level, 2^i coefficients are generated, and there can be as many as $\log_2 n$ levels. If the level number increases, this indicates that the data features are from a shorter time frame, and lower numbers stand for data features from a longer time frame.

2.2. Unequal-length segmentation

Unequal-length time series segmentation can be explored by using two approaches. The first approach aims to find specific data from numerous time series data, which are then treated as turning points that connect different segments. Pratt and Fink [23] proposed a turning point selection method, which treats these points as the maximal or minimal value of nearby data. Man and Wong [20] suggested another approach, where a function is used to measure the importance of every data point in a time series, and such importance becomes the maximal or the minimal value in a particular area. Data with high importance are then treated as transition points.

The second method solves the problem from a statistical point of view: data with similarities are separated from other data. The simplest method is to use a linear regression method to describe time series. There are many methods to segment this type of time series [18]. The Top-Down method [16] treats the entire time series as a whole, where it finds the division points with binary methods in order to divide the time series into segments. The Bottom-Up method [16] decomposes all data into the smallest units and then combines every two until a threshold value is reached. Another approach, called Sliding Windows [16], uses a fixed-width window to move around the time series. When the window reaches a data point with the upper segment limit cost, the point is then adopted as the cut-off point. According to Keogh et al. [16], the Bottom-Up method results in the least deviation in error cost.

When the number of segments is known prior to time series segmentation, the Global Iterative Replacement (GIR) [12] can be used for segmentation. In the GIR approach, the cut-off point in each segment is randomly chosen in the beginning, and the cut-off point with the most deviation is omitted before re-selecting a cut-off point with the smallest deviation. The process is continually repeated until no cut-off point needs replacing.

3. Ant Colony Optimization algorithm (ACO)

3.1. Traveling salesman problem (TSP)

This research uses the traveling salesman problem (TSP) as an example to explain the ACO algorithm. The research for this algorithm was described by the TSP [5,6]. In recent years, there have been several different ACO implementations suitable for optimization where the different algorithms are explained in the in the recent Ant

Colony Optimization book [7]. For the purpose of describing the TSP, we use the original ACO algorithm [5,6] as a description. The solution system was called the ant system (AS). Proofs of convergence for the ant algorithms are developed and described in several literatures [1,8–10,27].

In the traveling salesman problem, its purpose is to find the city that would allow a salesman to travel among a total of N cities by means of the shortest traveling path. Each city can only be passed once and the last stop must be the starting point. This problem is a NP-Complete problem, and the time for the solution would increase exponentially as the number of cities increases. Suppose (N, E) is used to express the graph of a TSP, where N stands for the number of cities to be traveled and E stands for the edges that connect the cities (an edge has a city at each end), with the meanings of the other symbols listed as follows:

(1) M: number of ants used when finding solutions.
(2) D_{ij}: length of the edge connecting city i and city j.
(3) $b_i(t)$: at time t, number of ants staying in city i; $i = 1, \ldots, N$, and $M = \sum_{i=1}^{N} b_i(t)$.

3.2. Algorithm

When first applying the ACO to the TSP (at time t), each ant randomly chooses a city as a starting point and then uses a probability value in choosing the next city to visit. Such a value is calculated by a Greedy Heuristic function which prioritizes the next nearest city as well as considers the pheromone strength. When time is $t + 1$, each ant visits the city next to the starting point. Each ant maintains a data structure called the Tabu List, which records the path the ant takes. When an ant visits a city, that city is put at the end of the Tabu List. Every time an ant is ready for another city, it checks its Tabu List to make sure that it does not visit any city in the list. When time reaches $t + N$, all ants must finish the tour and return to the starting point, which is the end of a cycle. Every time a cycle is finished, the amount of pheromone an ant leaves is recorded in each path, which also indicates the number of ants that have walked a particular path. The total strength of the pheromone on each path is also recorded. This algorithm is repeated until all ants are on the same path, or until the number of cycles reaches the user's preset threshold value.

3.3. Pheromone update method

Before the end of each cycle under the ACO algorithm, the residual pheromone on each connective path is re-calculated. The update method for the pheromone is described below. However, before we go any further, several variable definitions must first be understood:

(1) τ_{ij}: at time t, the pheromone left on a connective path (i,j).
(2) $\Delta\tau_{ij}$: at time t and time $t + N$, the amount of pheromone increased by all ants on a connective path (i,j).
(3) $\Delta\tau_{ij}^{m}$: between time t and $t + N$, the strength of pheromone left by the mth ant on a connective path (i,j).

From the above definitions, it is understood that the extra strength of the pheromone on a particular path can be determined by the sum of the pheromone left by each ant.

$$\Delta\tau_{ij} = \sum_{h=1}^{M} \Delta\tau_{ij}^{h}. \tag{1}$$

The pheromone on a connective path (i,j) left by the mth ant is the inverse of the total length traveled by the ant in a particular cycle. The formula is as follows:

$$\Delta\tau_{ij}^{m} = \begin{cases} \dfrac{Q}{L_m}. \\ 0 \end{cases} \tag{2}$$

In the above formula, Q is a constant, and L_m is the total path cost traveled by the mth ant during a cycle. Therefore, the longer an ant travels, the weaker the pheromone strength becomes on each connective path. In contrast, the shorter an ant travels on a path, the stronger the residual pheromone becomes. Therefore, the ants that travel more segments leave less pheromone.

Pheromone strength on a particular path increases with the number of ants that have traveled on it. However, the pheromone diminishes and disappears as time elapses. Variable $\rho(0 \leqslant \rho \leqslant 1)$ can be used to determine the residual strength of pheromone after decay between 2 cycles, and $1-\rho$ stands for the portion of pheromone that diminishes with time. Therefore at $t+N$, the residual pheromone is exactly the diminished amount plus the residual pheromone strength left by all the ants from the last cycle. The formula is as follows:

$$\tau_{ij}(t+N) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}. \tag{3}$$

Here, the method on how ants choose their paths is introduced. At the beginning of the algorithm, an ant located in city i must choose city j as the next city to visit. Because the pheromone strength on every path is still equal, there is no particular method of choice in picking the next city to visit. When ant k in city I chooses city j as the next city to visit, it will first check the Tabu List. Cities that are already listed are not allowed to be re-visited and the unvisited cities not in the list are grouped as $allowed_m$. Under the guidance of the Greedy Heuristic method, the closer each city j in $allowed_m$ is to city i, the stronger the pheromone on the connective path (i,j) is. The probability of the mth ant choosing city j as the next city to visit then becomes higher, with the probability function of visiting as follows:

$$p_{ij}^{m} = \begin{cases} \dfrac{\tau_{ij}(t)^{\alpha}\cdot\eta_{ij}^{\beta}}{\sum_{m\in allowed_k}\tau_{im}(t)^{\alpha}\cdot\eta_{im}^{\beta}}, & \text{if } j \in allowed_m. \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

In the formula, $\eta_{ij} = 1/D_{ij}$ is the inverse between connective paths (i,j), while α and β are the relative importance of pheromone strength and the distance between cities that affect an ant's judgment when choosing the next city to visit. When $\alpha = 0$, then the pheromone strength has no effect on the path choice, which means that the Greedy Heuristic method is enough to carry the calculation process. But according to Dorigo et al. [6], when solving TSP, $\alpha = 1$, $\beta = 5$, $\rho = 0.5$ would generate better results in the calculation process.

4. Time series segmentation algorithm

In this research, the four main parts in the time series segmentation algorithm are listed as follows:

(1) design of the visualization of problems,
(2) design of the visiting method and satisfying condition,
(3) design of cost functions and pheromone strength update function,
(4) design of updating visiting probability.

4.1. Problem definition and graph representation

We now have a time series S with time duration N, t functions as an individual index in the time series, $t = 0,\ldots,N-1$. At each time t, the time series data value is x_t. When each datum is arranged by time sequence, a complete time series $S = \{x_0, x_1, \ldots, x_{N-1}\}$ is formed. The segmentation of the time series divides the series into K segments, and each datum within the series can be assigned to a particular segment S_k. $f_k(t)$ is then used as a data model to describe the particular data in the segment. To simplify the comparison among current segmentation methods, every data model $f_k(t)$ in the segment is assumed to be a linear regression model. Under certain segmentation, the data value can be expressed as

$$E(x_t) = a_0 f_0(t) + a_1 f_1(t) + \cdots + a_{K-2} f_{K-2}(t) + a_{K-1} f_{K-1}(t). \tag{5}$$

The value of a_k is 0 or 1, $k = 0 \ldots K-1$. $a_k = 1$, if and only if data point x_t belongs to segment k.

To illustrate the time series by means of graph representation, a two-dimensional model is used to reflect each datum onto a particular point in the space. The x axis stands for time t, which is the index of the data, and the y axis stands for data x_t. After plotting, the two-dimensional space shows a diagram formed by N points that are arranged by index sizes, as demonstrated in Fig. 1.

Therefore, the rationale behind applying the ACO algorithm in time series segmentation is to find a path with the lowest cost from N points between x_0 and x_{N-1}, in order to connect all

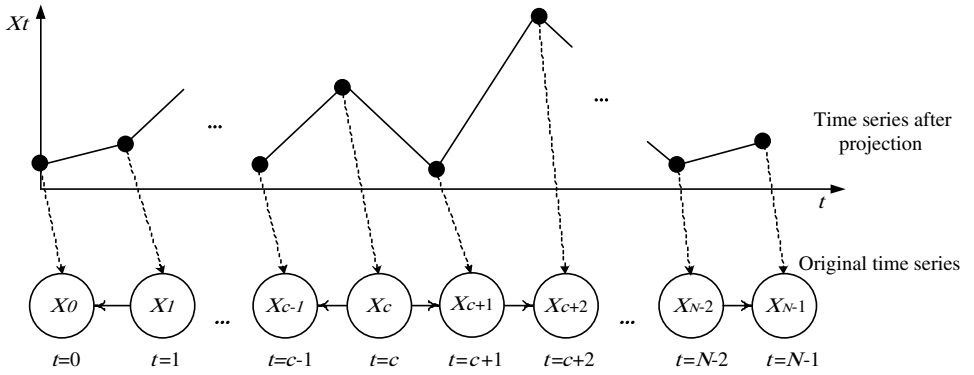

Fig. 1. Time series data and projection.

points. The entire route will pass $K-1$ points to divide the entire time series into K segments. Every time an ant reaches a point, this point will be treated as the end of a new segment. However, prior to determining what path to take, an ant can visit any point it wishes to, so any ant may be on any point and walk any path that connects its current city to the next. This is presented in the following, with Fig. 2 showing the illustration.

(1) There is a connecting point between every two points.
(2) Every connective path is one-way, which only allows a point with a smaller index value to lead to a point with a larger index value.

4.2. Visiting method and satisfaction of conditions

In this research, every ant moves from point x_i to x_j by choice based on probability. This means that the ant divides the data between $i+1$ and j into a segment, $S(x_{i+1}, x_j)$, except when $i=0$, in which the ant divides the segment into $S(x_0, x_j)$. When an ant completes a visit and reaches its finishing point x_{N-1}, it completes the time series segmentation. The visiting step and algorithm is listed in the following:

(1) An ant leaves point x_0.
(2) When an ant reaches point x_i, it chooses the next point to visit according to probability, and the probability is calculated based on the pheromone strength on each path.
(3) An ant continuously moves towards points with higher index values, and it stops visiting only when the index value equals $N-1$.
(4) When an ant finishes a visit, all points passed are recorded, and the cost function is used to calculate the total cost spent in finishing the routes.

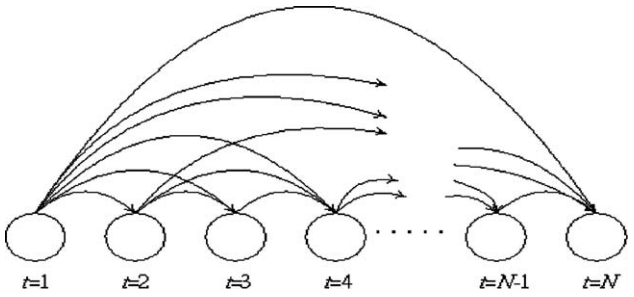

Fig. 2. The traveling route of the ACO algorithm.

(5) When an ant finishes all routes, it is considered a cycle. At the end of a cycle, all pheromones are updated according to the costs.
(6) The above steps are repeated until a threshold value is reached.

To stop the above algorithm, some threshold values must be set. In this research, there are several possible conditions that can be used to stop the algorithm.

(1) A completed cycle number is expressed by *NC*. When an algorithm progresses toward the *NC*th cycle, it will stop altogether at completion.
(2) The total cost of the paths is expressed by *TC*. During the algorithm, if the cost incurred by time series segmentation is found to be smaller than *TC*, then the search stops. This means that *TC* is treated as the highest value accepted by the user.
(3) All ants walk the same path, which means that, when path searching stagnates, the algorithm stops. This cuts path selection down to a minimum in the algorithm, which becomes impossible to break. It then reaches the local or global optimal solution.

4.3. Cost function and pheromone update function

When an ant finishes a visit, the cost of segmentation is calculated. In this research, data in the time series segments is treated as time series data generated by a specific model. Therefore, whether the data between the two points where an ant travels needs to be segmented individually is determined by the fitness between time series data and the data model. If the fitness between data and the data model is low, then the cost is high. This means that the segmentation of the time series data on the particular visit is inappropriate.

However, if only fitness is considered between the data and the data model, then time series may be over-segmented. For example, during linear regression, if the data is segmented at every two points, then a straight line passing between the two points can be calculated with no deviation. To avoid over-segmenting, whenever pheromone strength increases, another factor must be considered, which is the number of segments. For the above reasons, the pheromone strength function is designed as

$$\Delta\tau_{ij}^{m} = \frac{Q}{K_m^{\gamma} \cdot L_m}. \tag{6}$$

In the above formula, Q is a constant. Ants with high cost leave less pheromone on each connective path. In contrast, ants with less cost leave more pheromone on the paths. K_m is the number of segments the mth ant travels within the cycle. L_m is the total cost an ant travels within a cycle, which can be any function used to measure fitness between the actual data and the model used over the specific segment. For instance, simple linear regression can be calculated by mean squared error (MSE) or by sum of squared error (SSE). If MSE is adopted as the cost function, the calculation is

$$L_m = \frac{1}{N}\sum_{t=0}^{N-1}(x_t - E(x_t))^2. \tag{7}$$

In the formula, x_t is the actual value of the data at time t. $E(x_t)$ is the data model according to the segment where the data is located under the visiting method. This is the expected value calculated by a probability distribution function based on simple linear regression.

Aside from this, r is the only variable that requires user input, and can replace the number of segments or the cost threshold values that the current time series segmentation needs. It is a constant number between 0 and 1 ($0 \leqq r \leqq 1$), and it adjusts the affecting factors of segment numbers according to costs. When $r = 0$, pheromone reset only considers the strength of L_m, when $0 < r \leqq 1$, it means K_m must be considered in updating the pheromone. The greater r is, the better the effect of time series with small segments will have on pheromone strength. This means that r can be used to control the width of segmentation. The smaller r is, the higher the number of segments become. Therefore, the time series pattern obtained would fit individual information better, yet the pattern would weaken in reflecting the total

data features. The higher r is, the lower the number of segments become, and the fitness between pattern and individual data weakens.

4.4. Update method of visiting probability

According to previously described cost functions, whenever a cycle ends, this algorithm can re-calculate the residual pheromone strength on every path.

$$\tau_{ij}(c) = \rho \cdot \tau_{ij}(c-1) + \Delta\tau_{ij}. \tag{8}$$

c is the index value of cycle numbers, and $\rho(0 \leqslant \rho \leqslant 1)$ is the ratio that controls the remaining strength of the gasified pheromone in two cycles. Every time a cycle ends, the residual strength of the pheromone on a particular path becomes the residual pheromone strength plus all the pheromone left on the path by all ants in the cycle.

After balancing inspirational method and cost function calculation, the visiting probability update can be calculated as the following:

$$p_{ij} = \begin{cases} \frac{\tau_{ij}}{\sum_{l=i+1}^{N} \tau_{il}}, & \text{if } j > i, \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

4.5. Similarity measurement between time series

Dynamic Time Warping (DTW) was first suggested by Sakoe and Chiba [25]. This similarity measurement of the time series was widely promoted and improved upon. Many scholars used this method to cluster time series data [2,3] as well as measure the similarity among time series data of different lengths [14]. The researchers proved that dynamic time warping has excellent effect on time series applications. Therefore, DTW is also in this research to measure the similarities among the time series data.

Although DTW can be used to measure the similarities or distance between two time series, this method can only be used under one pre-condition, and this is that the element distances in time series must be measured in a numeric way. Therefore, prior to using DTW for measuring similarities, there must be a numeric measurement for distances between segments. However, there are different segment features under different time series data expressions, so it is thus required to have different distance-measuring methods. This research is continued by designing a segment distance measurement method.

4.5.1. Segment distance measurement

When using the ACO algorithm for time series segmentation, optimized simple linear regression lines are obtained to minimize the segmentation cost. Traditionally, a simple linear regression line has two coefficients: the line's slope and its intercept. However, the intercept of a simple linear regression line is not as meaningful as the average value from the segment data. In calculations, the data average generally falls on the center of a simple linear regression line. This is an even easier and more direct for average value calculation than the intercept value.

In addition, the length of each segment, the deviation of data and the data model distinguish each segment. Two feature values may therefore be added to each segment—segment length and data deviation. In this research, the features of each segment consist of four elements: length, sum of squared error (SSE) for representing data deviation, average data value, and the slope of a simple linear regression line. When assigning the feature values above, the simplest method for measuring segment distance is by using the Euclidean distance.

Given that Q_i and C_j are the ith and jth segment in time series Q and C. $\varsigma_1(Q_i)$ and $\varsigma_1(C_j)$ represent the average value in each segment, respectively, while $\varsigma_2(Q_i)$ and $\varsigma_2(C_j)$ represent the slopes of respective simple linear regression line in the data model. $\varsigma_3(Q_i)$ and $\varsigma_3(C_j)$ are the SSE values in the two segments. Finally, $\varsigma_4(Q_i)$ and $\varsigma_4(C_j)$ represent the length of each segment. The importance of these four feature values might vary in different problems. Therefore, weight parameters of ω_1, ω_2, ω_3 and ω_4 have been added to this research to control the relative importance of these four feature values. After defining the above parameters, $d(Q_i, C_j)$ can then be measured with the following formula:

$$d(Q_i, C_j) = \sum_{h=1}^{4} \omega_h \sqrt{[\varsigma_h(Q_i) - \varsigma_h(C_j)]^2}. \tag{10}$$

4.5.2. Dynamic time warping calculation

After obtaining the distance between segments, it becomes much easier to calculate the distance between time series by DTW. The calculation is conducted as follows:

(1) Calculate iteratively by dynamic programming: Conduct dynamic programming iteratively for calculating the minimal cumulative distance between segment Q_i and C_j. It is done by using the following formula.

$$\gamma(i,j) = d(Q_i, C_j) + \min\{\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1)\}. \quad (11)$$

(2) Calculate the distance between time series Q and C: If Q and C each has n and m segments, respectively, the distance can be expressed by using the following formula.

$$\mathrm{DTW}(Q, C) = \gamma(n, m) \in [0, \infty]. \quad (12)$$

When using the DTW algorithm with recursion to calculate the similarity between time series, the required time complexity is O(nm).

5. Experiment design and results

In the original ACOS algorithm, time series is arranged from left to right. This reveals two problems. One is that, at the beginning, the insufficiency of segmentation makes segments with large differences undistinguishable. The other problem is that data at the end are over-segmented, so that similar data are segmented into several segments. This segmentation result is mainly because each ant travels from the first point in the ACO algorithm. Under this circumstance, ants are given many choices on which city to visit next when they are at the front end of the time series data. Yet, when ants approach the end of the time series, the points for selection are very limited. Thus, each selection point has a high probability of being chosen as the next point to visit. Even cost function and pheromone update calculation can solve the dilemma in this circumstance, moving to the correct selection point as the next choice of visiting with limited cycles is not easy, especially if the visiting method has a low cost. Therefore, this is already the best local optimal solution in a dilemma area.

To solve this problem, this research slightly modifies the original ACOS algorithm. In the new algorithm, ants do not have to start in the first data point, but can randomly start at any point. This modified algorithm is called the Random ACOS algorithm (abbreviated as RACOS in this research), which is further detailed in the following section.

5.1. Re-designing the algorithm

The biggest difference between the RACOS algorithm and the ACOS algorithm is the selection of the starting point. In ACOS, ants in each cycle all start from the first data point, and end at the end point of the time sequence. In RACOS, each ant selects a random point as a starting point and then visits the rest of the points in two stages. In stage one, an ant can move towards the end of the data and finish at the last data point. In stage two, it moves toward the beginning of the data until the first is reached. This means that the above two-direction visits can connect at the starting point. Fig. 3 demonstrates the different routes taken. Under this visiting route, the steps of this algorithm are:

(1) An ant picks a random point x_c as the starting point.
(2) In the first stage, the ant moves toward the end of the time series. When the ant is at point x_i, it then chooses the next point to visit, x_j, by probability P_{ij}, given $j > i$.
(3) Next, the ant continues to move towards points with larger index values until the point it is at equals $N - 1$, wherein the ant stops visiting.
(4) In the second stage, the ant moves toward the beginning of the data. When it is on x_j, it then chooses the next point to visit, x_i, by probability P_{ij}, given $j > i$.
(5) The ant continues to move towards points with lesser index values until the point it is at equals 0, wherein the second stage would then be completed.

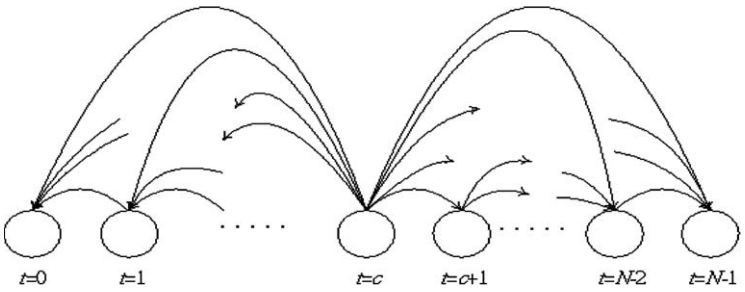

Fig. 3. Random ACOS visiting path selection.

(6) An ant has completed the entire route once the two stages are finished, with the visited points recorded. Of course, included is the starting point x_c. After, the cost function is applied to calculate the total cost of that path.
(7) Each time an ant completes a visit, the cycle ends. The cost on each path is then calculated to get the update of pheromone on the path.
(8) The above steps are repeated until the threshold value is reached.

5.2. Experiments of real data

Now that the modified algorithm has been demonstrated, this research will identify the differences between the RACOS algorithm, the ACOS algorithm, and the Bottom-Up method.

5.2.1. Experiment data

In the real world, most time series data do not show the number of segments. Therefore, at this stage of experimentation, we will use two real time series data for our experiments: the S&P 500 index and daily gold prices. In the S&P 500 index, 388 seasonal data of 97 years between 1900 ad 1996 are used, which is represented as "9_17b". In daily gold prices, 1074 daily gold price data between January 1, 1985 and March 31, 1989 are used, which is represented as "Gold". The data is obtained from online time series databases (http://www-personal.buseco.monash.edu.au/hyndman/~TSDL/).

5.2.2. Procedures

There are three major goals in this experiment. First, we aim to compare cost and accuracy when we conduct tasks through the RACOS, ACOS and Bottom-Up algorithms. Secondly, results generated by different input values of r are observed for the RACOS and ACOS algorithms. Lastly, since ACOS is an artificial intelligence approach, the results may vary even when the same data is segmented. We shall compare the stability of the results generated by the RACOS and ACOS.

To achieve such goals, this research inputs six different values for r: 0, 0.2, 0.4, 0.6, 0.8, and 1, for time series 9_17b and Gold. The segmentation costs and number of segments are recorded. Each value r runs ten experimentations to stabilize the results. Such stability shows the relative distribution of the generated segments under different segmentation methods, with different input values for parameter r. This relative distribution is measured by coefficient of variation (CV), calculated as follows:

$$\mathrm{CV}_{\mathrm{ACOS}} = \frac{s^{(K)}_{\mathrm{ACOS}}}{\mathrm{Mean}^{(K)}_{\mathrm{ACOS}}}, \tag{13}$$

$$\mathrm{CV}_{\mathrm{RACOS}} = \frac{s^{(K)}_{\mathrm{RACOS}}}{\mathrm{Mean}^{(K)}_{\mathrm{RACOS}}}. \tag{14}$$

$\mathrm{CV}_{\mathrm{ACOS}}$ stands for the coefficient of variation, while $s^{(K)}_{\mathrm{ACOS}}$ stands for the standard deviation of the segment number from running the ACOS algorithm iteratively. $\mathrm{Mean}^{(K)}_{\mathrm{ACOS}}$ is the mean value of the segment numbers. When $\mathrm{CV}_{\mathrm{ACOS}}$ is greater than $\mathrm{CV}_{\mathrm{RACOS}}$, the segment number generated by the ACOS algorithm has a greater level of rel-

ative dispersion than that of the RACOS algorithm. Therefore, the former tends to have less stable segment numbers while the latter can have less stable segment numbers.

In the Bottom-Up method, the segment numbers generated by each of the above methods are used as threshold values. Thus, the cost comparison can be performed under identical segment numbers.

5.2.3. Experiment results

5.2.3.1. Cost comparison. To measure the accuracy using different r inputs, segment numbers which range from 1 to 31 are generated by the RACOS and ACOS. Fig. 4 shows the comparison of the two sets of data with equal segment numbers generated. From the figure, it is obvious that when the number of segments is low, the Bottom-Up method results in a highest cost. But when the segment number is high, the ACOS results in the highest cost in data 9_17b. The ACOS algorithm obviously has the highest cost compared to the two other methods, while costs and segment numbers from the RACOS are similar to the Bottom-Up method.

Dorigo and Stützle [7] describe and explain several recently developed modified ACO algorithms. According to [7], the modified ACO algorithms outperform the original ACO algorithm in optimization. Therefore, from the above experiments and comparisons, we believe that we can easily improve these results by implementing a state-of-the-art ACO algorithm, like the MAX-MIN Ant system [28], for example.

5.2.3.2. Selection of values of parameter r. In the time series data experiments of 9_17b and Gold, each parameter r was repeated ten times, and the segment numbers generated were recorded. The generated segment numbers were then averaged and presented in Fig. 5. In Fig. 5, it is discovered that a different segmentation method produced significant effects on parameter r. For data 9_17b, when r is set at 1, the mean value of segment numbers generated by the ACOS is 1, and 5 for the RACOS. Obviously, when the same value of r is used, the segment numbers generated by the RACOS are higher than those of the ACOS.

5.2.3.3. Relative stability. As mentioned above, the evaluation of algorithm stability focuses on the segment number stability generated by the segmentation algorithms. Fig. 6 shows that the variation coefficient of the ACOS is significantly larger than the variation coefficient of the RACOS. Exceptional cases occur when the r input is 1, since the calculation of coefficient of variation comes from the standard deviation being divided by the mean value.

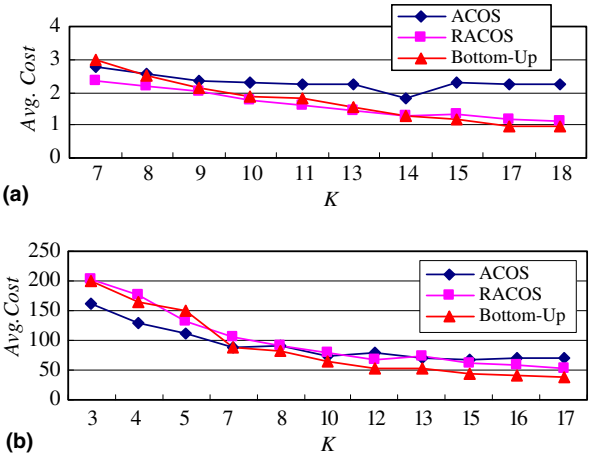

Fig. 4. Cost comparison from real data experiments.

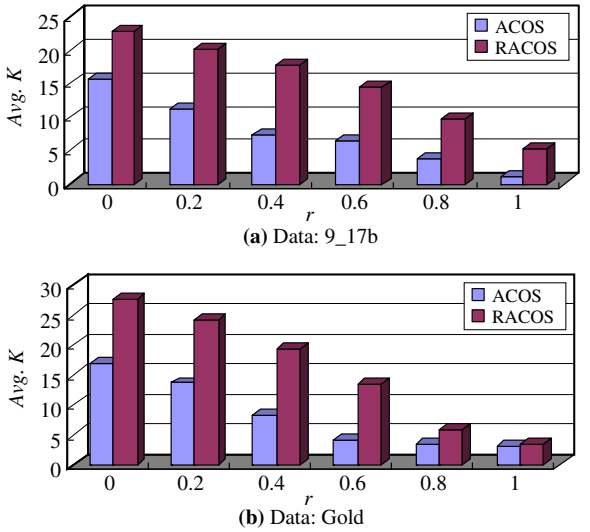

Fig. 5. The effect of r on segment numbers from real data.

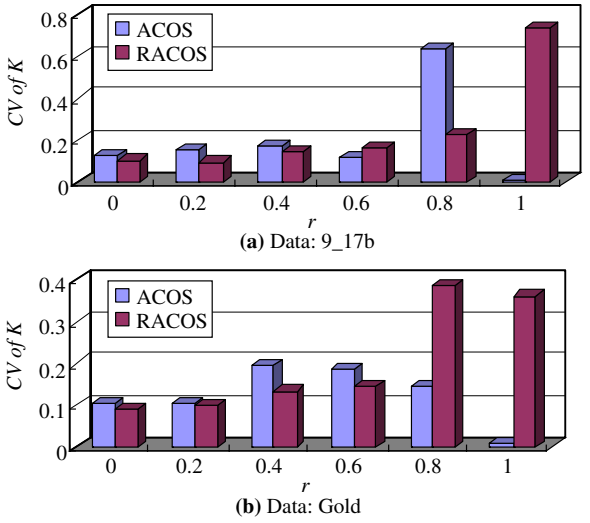

Fig. 6. Comparison of coefficient of variation on segment numbers.

For data 9_17b, when r is 1 during the ACOS process, the generated segment numbers have the values of 1, which means the entire time series is treated as an undivided whole. Under this circumstance, the ten repeated experiments all generated 1 as the resulting segment numbers. Therefore, the standard deviation is 0. The coefficient of variation is still 0 after calculation. Therefore, it is fair for us to conclude that it is meaningless to use 1 for r. Yet, in data Gold, when r is set as 0.8, the same situation occurs. Thus, 1 is not a suitable value for r in most circumstances.

Fig. 6 shows that the smaller r is, the more segment numbers are generated. Therefore, its relative stability becomes higher. In data 9_17b, when seg-

mented by the RACOS, and r is set to 0.2, the average number of segments is 20 with 1.89 for the standard deviation. When r is set at 0.8, the average segment number is 9.7 with the standard deviation of 2.26. When the value of r is small, the average segment number not only increases, but the standard deviation also decreases with the increased average segment number. Therefore, when r is small, stability increases.

With these three experiment results, it is apparent that the modified algorithm has differences from the original ACOS algorithm, and they are depicted as follows:

(1) When the same value of r is used as an input parameter, the time series segment number is higher.
(2) Regardless of r or data length, the segmentation result for the RACOS has more stability than the ACOS.
(3) The cost is less during segmentation.

Lastly, in calculating the required times, the starting point selection is different for the RACOS algorithm and the ACOS algorithm, though the time required result is almost the same.

5.3. Experiment of measuring similarity

As mentioned at the beginning of the research, a good time series data segmentation algorithm must be able to keep the original information and avoid data loss during segmentation, so that it will retain high data quality for data mining and other analysis. The experiments at this stage will continue from the previous experiments, and data retention ability is compared between the RACOS, which has lower segmentation cost, and the Bottom-Up method.

5.3.1. Experiment data

To measure data loss during time series data segmentation, this research adopts an enormous amount of real data for experimentation. The data used in this section are the records of daily closing prices of 52 financial companies in the year 2000. The average data length is 277, with the shortest being 25.

5.3.2. Procedures

At this stage, we aim to measure the degree of data loss in different time series segmentation methods. To achieve this purpose, this stage of research focuses directly on the ACOS and Bottom-Up methods to calculate the distance between the time series data through dynamic time warping. The hypothesis of this stage of the experiment is that, if a segmentation method can result in less data loss, the time series distance after segmentation with that method will have a certain association to the distance of the original data. Under this hypothesis, this stage of research then calculates the distances between the paired data of the 52 original time series data by dynamic time warping. This means that, while having 52 time series data, the distances must be calculated between after every two time series, which would generate $C_2^{52} = 1326$ distance data. After, different segmentation methods are conducted on the original data. Similarly, when each segmentation task is completed, the segment parameters can then be used to calculate the distance between each pair of the time series. This research adopts a correlation coefficient to calculate the distance association between time series data. Given a set of observations $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, the formula for computing the correlation coefficient becomes:

$$\gamma_{xy} = \frac{1}{n-1} \sum \left(\frac{x - \bar{x}}{S_x} \right) \left(\frac{y - \bar{y}}{S_y} \right). \tag{15}$$

In the formula (15), S_x and S_y are the standard deviations of x and y, respectively. The correlation coefficient always takes a value between -1 and 1, with 1 or -1 indicating perfect correlation. A positive correlation indicates a positive association between the variables (increasing values in one variable correspond to increasing values in another variable), while a negative correlation indicates a negative association between the variables (increasing values in one variable correspond to decreasing values in another variable). A correlation value of 0 indicates no association between the variables. The segmentation methods mentioned above are the Bottom-Up and RACOS algorithms. Four values of r: 0.2, 0.4, 0.6 and 0.8, are used in the RACOS. During the experiment, it is discovered that time series data is

divided into ten to twenty segments. We have identified 10, 12, 14, 16, 18, and 20 as the 6 threshold values in the Bottom-Up method.

Lastly, there are four parameters: ς_1, ς_2, ς_3, ς_4, to describe the features of the segments: a segment's average value of data, the slope, the sum of squared errors, and segment length. The formula is

$$d(Q_i, C_j) = \sum_{h=1}^{4} \omega_h \sqrt{[\varsigma_h(Q_i) - \varsigma_h(C_j)]^2}. \quad (16)$$

To observe the effects on distance measurements on these four parameters, this research also tries several different parameter weightings to observe and record the distance relationship of the time series data after segmentation. For example, ω_1, ω_2, ω_3, and ω_4 are used to represent the weighting setup, and there are six sets of weights adopted in this experiment, five of them with constant weights being $(1,0,0,0)$, $(0,1,0,0)$, $(0,0,1,0)$, $(0,0,0,1)$, and $(1,1,1,1)$. The aforementioned weighting setups are used to measure the segment data average, the slope, the sum of squared error, and level of contribution of segment length on distance, all under different segment methods. The fifth set of weights is used to measure the association of distances in the time series data in the segments and original data when four feature weights are set to be equal.

5.3.3. Experiment result

For other inputs of r in the RACOS, the number of segments is recorded and displayed in Table 1. This table corresponds to the previous results, which is that different r input can affect the output segment numbers determined by the RACOS, but there are no significant changes in the standard deviation.

Table 2 is the comparison table of correlation coefficients where the distances of two time series data are calculated by dynamic time warping before and after segmentation from the original data and from the results of different segmentation methods. Each row in the table represents the difference of the relative coefficient from the different segmentation methods and parameter inputs. In the Bottom-Up method, the parameter is the target segment number. In the RACOS algorithm, the parameter is the different inputs of r. Each field represents the relative coefficient results of different feature values in each segment.

- $(1,0,0,0)$ is the feature value that adopts the average value in each segment as distance measurement.

Table 1
Segment number outputs using the RACOS

r	Average value	Standard deviation
0.8	11.30769	2.81131
0.6	14.46154	1.88348
0.4	17.13462	2.70822
0.2	19.26923	2.57526

Table 2
Comparison of correlation coefficients for distance measurement

Method of segmentation	Parameter	$(1,0,0,0)$	$(0,1,0,0)$	$(0,0,1,0)$	$(0,0,0,1)$	$(1,1,1,1)$	$(\rho_1,\rho_2,\rho_3,\rho_4)$
Original data	–	1.0	–	–	–	–	–
Bottom-Up	10	0.9476	0.8169	0.8187	0.2010	0.8393	0.8515
	12	0.9446	0.8295	0.8167	0.3362	0.8464	0.8580
	14	0.9455	0.8281	0.8088	0.3969	0.8500	0.8622
	16	0.9512	0.8321	0.8122	0.3340	0.8634	0.8770
	18	0.9573	0.8409	0.8298	0.1808	0.8872	0.9020
	20	0.9619	0.8556	0.8397	0.1346	0.9030	0.9178
RACOS	0.8	0.9647	0.8072	0.7790	0.0245	0.7671	0.8674
	0.6	0.9769	0.8590	0.8384	0.0211	0.8863	0.9040
	0.4	0.9847	0.8726	0.8451	−0.0092	0.9050	0.9241
	0.2	0.9827	0.8539	0.8374	0.0127	0.9029	0.9206

- $(0,1,0,0)$ is the feature value that adopts the segment slope in each segment as distance measurement.
- $(0,0,1,0)$ is the feature value that adopts the sum of the squared error in each segment as distance measurement.
- $(0,0,0,1)$ is the feature value that adopts the segment length for each segment as distance measurement.
- $(1,1,1,1)$ considers all average values, slopes, sum of squared errors, and segment lengths with equal importance.

$(\rho_1,\rho_2,\rho_3,\rho_4)$ also considers the above four features when measuring time series distances, yet the importance of each feature is based upon the correlation coefficient between distance measurement considered by that feature and the original data distance measurement.

According to the results listed in Table 2, the effect of the calculation of distance after time series segmentation can be interpreted in two ways. First, from the viewpoint of segment features, we can see that in the four features, when distances are measured by their segment data average, the effect is most similar to using original data for distance measurement. Thus, we can conclude that this data loss is the lowest. The effects of using the slope feature and the sum of squared error are similar, and the correlation coefficients are not as high as the average value result, but are still 0.8 or higher. However, the segment length has the lowest correlation coefficient, even having negative correlation. Therefore, using the segment length as the feature value for distance measurement causes the largest data loss. Lastly, when all four features are considered, the correlation of distance measurement is not as good as when the average value is considered. Yet Table 2 demonstrates that its effect is as high as second or third place. However, when all four features have the same weights, the resulting correlation cannot compete with the approach when weights are set according to correlation coefficients.

However, in looking at the two setting methods of weights with the highest correlation, the highest is $(1,0,0,0)$, and the second is $(\rho_1,\rho_2,\rho_3,\rho_4)$. In these two instances, the RACOS can generally generate better correlation than the Bottom-Up method. In other words, the RACOS can better prevent data loss in time series segmentation.

6. Conclusion

Living in an era of explosive data and information, people generate and accumulate data daily. Many transaction data details each movement and exchanges, yet the speed of accumulation makes understanding them within a limited time difficult. Thus, many calculation methods are developed to integrate and combine data to present approximate figures. For example, online analytical processing (OLAP) is a tool for such a purpose. However, although many calculation methods enable people to understand the environment information from data, it is still difficult to analyze or extract useful knowledge from temporal behaviors.

For this reason, this research aims to discuss the matter from the perspective of time series data in order to pursue a real quality-promoting data mining method for time series data. However, time series data and static data vary in nature. Thus, much time, effort and cost are required for calculation. Our experiments have proven that the risk of data loss is lowered, which demonstrates our application's ability in promoting data quality during data analysis or data mining.

In addition, this research applies artificial intelligence approaches to the time series data mining. It simplifies the segmentation process and reduces parameter inputs, which are the conceptual parameters for fineness of segmentation. As opposed to other segmentation requirements of threshold values (such as segment number, upper limit of total cost, upper limit of segment cost), this approach does not require pre-understanding original data such as domain range, data distribution, number of segments, etc. Through the experiments, the results of segmentation are seen to be similar or even better than those of the currently popular Bottom-Up method.

To highlight the purpose and target of the experiments, several limits are considered:

(1) Limit of simple linear regression
To make calculation convenient, simple linear regression is used to describe segments of the time series data. In reality, some data cannot be described by simple linear regression [26], and such application might result in undesirable segmentation results.
(2) Batch handling
The ACO algorithm is a solution-finding process; therefore, multiple accesses and calculation processes on time series data are necessary for better results. The methods proposed in this research handle data by batches and do not consider online calculations.
(3) Providing acceptable solution
The ACO algorithm is an unsupervised solution-finding method, and the answers it provides are not guaranteed to be the best, but the expected effects can reduce human involvement with an acceptable solution obtained through automatic or semi-automatic methods.

Time series segmentation is only the first step in the processes of time series data mining. Distinguishing the transition process of time series data better enables one to extract useful information from an enormous time series database. Yet, how to effectively use the information or transform it into useful knowledge are issues that require further research. In conclusion, several future research directions are proposed for time series segmentation in data mining.

First, to a certain degree, the purpose of time series data segmentation is to re-arrange data into simple sequential data. After segmentation, data in the same segment is treated as values of observation with the same status. This makes it possible, by means of observation, to analyze whether a particular data pattern will repeatedly reappear from time to time—a task of data mining towards the periodic time series patterns. On the other hand, analyzing the arrangement patterns of different data models can also be used to conduct data mining. With this application, causal relationships can be found for time series data or sequential data. Such use of data mining can draw associations of particular events from data, and even predict occurrences of events. Of course, prevention is another action that can be utilized if such occurrences are not desired.

Furthermore, if there are many time series data in databases, data mining can be applied to an even greater degree. For many strings of times series data, one important problem is how to measure the time series similarity. Time series segmentation and Dynamic Time Warping used in this research are methods used in distance measurement before and after segmentation, which address the problem. Thus it is possible to cluster or categorize time series data under the condition of being able to measure distances among different time series. For example, a telecommunications company, by recording customers' spending and usage, can analyze the customers' time series patterns in order to classify them according to behavioral patterns through time changes, which can help the company design future marketing projects or customer relationship strategies.

Acknowledgements

This research was partially supported by the National Science Council, Taiwan, ROC, under grant number NSC 93-2213-E-030-009. The authors also express their sincere gratitude to the anonymous referees for their helpful criticism and valuable suggestions.

References

[1] A. Badr, A. Fahmy, A proof convergence for ant algorithms, Information Sciences 160 (2004) 267–279.
[2] G. Chen, Q. Wei, H. Zhang, Discovering similar time-series patterns with fuzzy clustering and DTW methods, in: Proceedings of Joint 9th IFSA World Congress and 20th NAFIPS International Conference, Vancouver, 2001, pp. 2160–2164.
[3] S. Chu, E. Keogh, D. Hart, M. Pazzani, Iterative deepening dynamic time warping for time series, in: Proceedings of the Second SIAM International Conference on Data Mining, Arlington, 2002.
[4] M. Dorigo, G. Di Caro, L.M. Gambardella, Ant algorithms for discrete optimization, Artificial Life 5 (3) (1999) 137–172.
[5] M. Dorigo, L.M. Gambardella, Any colony system: A cooperative learning approach to the traveling salesman

problem, IEEE Transactions on Evolutionary Computation 1 (1) (1997) 53–66.
[6] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: Optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics 26 (Part B(1)) (1996) 29–41.
[7] M. Dorigo, T. Stützle, Ant Colony Optimization, The MIT Press, Cambridge, MA, 2004.
[8] M. Gavrilov, D. Anguelov, P. Indyk, R. Motwani, Mining the stock market: Which measure is best? in: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, 2000, pp. 487–496.
[9] W.J. Gutjahr, A graph-based ant system and its convergence, Future Generation Computer Systems 16 (8) (2000) 873–888.
[10] W.J. Gutjahr, ACO algorithms with guaranteed convergence to the optimal solution, Information Processing Letters 82 (3) (2002) 145–153.
[11] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, San Francisco, 2001.
[12] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmäki, H. Toivonen, Time series segmentation for context recognition in mobile devices, in: Proceedings of the IEEE International Conference on Data Mining, San Jose, 2001, pp. 203–210.
[13] K. Kalpakis, D. Gada, V. Puttagunta, Distance measures for effective clustering of arima time-series, in: Proceedings of the IEEE International Conference on Data Mining, San Jose, 2001, pp. 273–280.
[14] E. Keogh, M.J. Pazzani, Scaling up dynamic time warping to massive datasets, Lecture Notes in Artificial Intelligence 1704 (1999) 1–11.
[15] E. Keogh, K. Chakrabarti, S. Mehrotra, M.J. Pazzani, Locally adaptive dimensionality reduction for indexing large time series databases, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Santa Barbara, 2001, pp. 151–162.
[16] E. Keogh, S. Chu, D. Hart, M.J. Pazzani, An online algorithm for segmenting time series, in: Proceedings of the IEEE International Conference on Data Mining, San Jose, 2001, pp. 289–296.
[17] E. Keogh, K. Chakrabarti, M.J. Pazzani, S. Mehrotra, Dimensionality reduction for fast similarity search in large time series databases, Knowledge and Information Systems 3 (3) (2001) 263–286.
[18] E. Keogh, S. Lonardi, B. Chiu, Finding surprising patterns in a time series database in linear time and space, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, 2002, pp. 550–556.
[19] M. Last, Y. Klein, A. Kandel, Knowledge discovery in time series databases, IEEE Transactions on Systems, Man, and Cybernetics 31 (Part B(1)) (2001) 160–169.
[20] P.W.P. Man, M.H. Wong, Efficient and robust feature extraction and pattern matching of time series by a lattice structure, in: Proceedings of the 2001 ACM CIKM International Conference on Information and Knowledge Management, Atlanta, 2001, pp. 271–278.
[21] R.S. Parpinelli, H.S. Lopes, A.A. Freitas, An Ant Colony based system for data mining: Applications to medical data, in: Proceedings of the 2001 Genetic and Evolutionary Computation Conference (GECCO 2001), San Francisco, 2001, pp. 791–798.
[22] R.S. Parpinelli, H.S. Lopes, A.A. Freitas, Data mining with an Ant Colony Optimization algorithm, IEEE Transactions on Evolutionary Computing 6 (4) (2002) 321–332.
[23] K.B. Pratt, E. Fink, Search for patterns in compressed time series, International Journal of Image and Graphics 2 (1) (2002) 89–106.
[24] D. Rafiei, On similarity-based queries for time series data, in: Proceedings of the 15th International Conference on Data Engineering, Sydney, Australia, 1999, pp. 410–417.
[25] H. Sakoe, S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, IEEE Transactions on Acoustics Speech and Signal Processing ASSP-26 (1) (1978) 43–49.
[26] A.F. Sheta, K.D. Jong, Time series forecasting using GA-tuned radial basis functions, Information Sciences 133 (2001) 221–228.
[27] T. Stützle, M. Dorigo, A short convergence proof for a class of ACO algorithms, IEEE Transactions on Evolutionary Algorithms 6 (4) (2002) 358–365.
[28] T. Stützle, H.H. Hoos, MAX-MIN Ant system, Future Generation Computer Systems 16 (8) (2000) 889–914.
[29] G.P. Zhang, M. Qi, Neural network forecasting for seasonal and trend time series, European Journal of Operational Research 160 (2) (2005) 501–514.