SOFTWARE—PRACTICE AND EXPERIENCE
Softw. Pract. Exper. 2009; **39**:81–103
Published online 17 July 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spe.891

Case-based curve behaviour prediction

Vladimir Kurbalija*, †, Mirjana Ivanović and Zoran Budimac

Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, Trg D. Obradovića 4, 21000 Novi Sad, Serbia

SUMMARY

Case-based reasoning (CBR) is the area of artificial intelligence where problems are solved by adapting solutions that worked for similar problems from the past. This technique can be applied in different domains and with different problem representations. In this paper, a system curve base generator (*CuBaGe*) is presented. This framework is designed to be a domain-independent prediction system for the analysis and prediction of curves and time-series trends, based on the CBR technology. *CuBaGe* employs a novel curve representation method based on splines and a corresponding similarity function based on definite integrals. This combination of curve representation and similarity measure showed excellent results with sparse and non-equidistant time series, which is demonstrated through a set of experiments. Copyright © 2008 John Wiley & Sons, Ltd.

Received 1 June 2006; Revised 3 May 2008; Accepted 16 May 2008

KEY WORDS: artificial intelligence; case-based reasoning; time-series analysis; curve prediction

1. INTRODUCTION

Case-based reasoning (CBR) is a relatively new and promising area of artificial intelligence and it is also considered as a problem-solving technology (or technique). The CBR technology is used for solving problems in domains where experience plays an important role [1–5].

During the past 10 years, the CBR has become a successful technique for knowledge-based systems in different domains. This promising technique is based on the use of previous experience, which is stored in the form of cases, to better understand and solve new problems in a particular domain. The main idea is an assumption that, for many particular domains, similar problems usually have similar solutions.

*Correspondence to: Vladimir Kurbalija, Department of Mathematics and Informatics, Faculty of Science, University of Novi Sad, Trg D. Obradovića 4, 21000 Novi Sad, Serbia.
†E-mail: kurba@im.ns.ac.yu

Generally speaking, the CBR systems have to not only provide solutions to problems but also take care of other tasks that may occur when they are used in practice. Thus, CBR can mean different things depending on the intended use of reasoning:

- to explain current situation according to previously experienced similar situation;
- to critique current situation based on old cases;
- to reason from precedents to understand a current situation;
- to combine old solutions in order to solve a current problem;
- to obtain a complete solution or a part of it based on previous cases.

There has been a recent research at the Department of Mathematics and Computer Science (University of Novi Sad) in the area of implementing decision support systems using CBR. One decision support system called '*CaBaGe*' (case base generator) [6–8] had been implemented already. The main characteristic of this system was that it was domain independent: the input for the system was a database of previously solved problems (described by a set of attributes) and the system could solve new problems (described by a smaller set of attributes) using the solutions of similar problems from the past. The domain of the cases in the database was not important. Cases could be medical examinations, business data, results of experiments, etc., but it was important that all the cases in one database were from the same domain.

Recently, we defined some new goals for future decision support systems: in many practical domains some decisions depend on the behaviour of time diagrams, charts or curves. Therefore, it would be very useful to implement a system that could help experts in making their decisions by analysing curves, comparing them with similar curves from the past and may be predicting the future behaviour of a current curve on the basis of most similar curves from the past.

The system 'curve base generator' (*CuBaGe*) was the direct result of these intentions. The implemented system retrieves curves that are most similar to the current problem curve, from the database of previous curves. On the basis of the retrieved curves, the system can predict some future behaviour of the current problem curve. This system has been already applied successfully in predicting the rhythm of issuing invoices and receiving actual payments at the company 'Novi Sad Fair' [9,10].

In this paper, the decision support system *CuBaGe*, which has the purpose of predicting curve behaviour, will be presented. The remainder of this paper is organized as follows. The following section gives a brief introduction to the CBR technology. In Section 3, some problems concerning similarity between the curves are presented. Section 4 is dedicated to the implementation of the system *CuBaGe*, whereas Section 5 covers a real-world application of the system in financial prediction. In Section 6, some related work is mentioned. At the end, a conclusion and some future works are given.

2. FOUNDATIONS OF THE CBR TECHNOLOGY

Generally speaking, CBR is applied to solving new problems by adapting solutions that worked for similar problems in the past. The main hypothesis here is that similar problems have similar solutions. The basic scenario for most of the CBR applications is the following: In order to find a solution to an actual problem, first look for a similar problem in an experience base, then take the solution from the past and finally use it as a starting point to find a solution to the actual problem.

DOI: 10.1002/spe

In CBR systems, experience is stored in the form of cases. A case is a recorded situation where a problem was completely or partially solved, and it can be represented as an ordered pair (*problem*, *solution*). The entire experience is stored *in the case base*, which is a set of cases in which a case represents a previous episode where the problem was successfully solved.

The main problem in CBR is to find a good similarity measure—the measure that can tell to what extent two problems are similar. In functional terms, similarity can be defined as a function

$$sim : U \times CB \rightarrow [0, 1] \tag{1}$$

where U refers to the universe of all objects (from a given domain), whereas CB refers to the case base (only those objects that were examined in the past and saved in the case memory). A higher value of the similarity function means that these objects are more similar.

The CBR system has the goal of not only providing solutions to problems but also of taking care of other tasks that occur when it is used in practice. The main phases of the CBR activities are described in the *CBR-cycle* (Figure 1) [2].

In the *retrieve* phase, the most similar case (or k most similar cases) to the problem case is retrieved. In the *reuse* phase, some modifications to the retrieved case are made in order to provide a

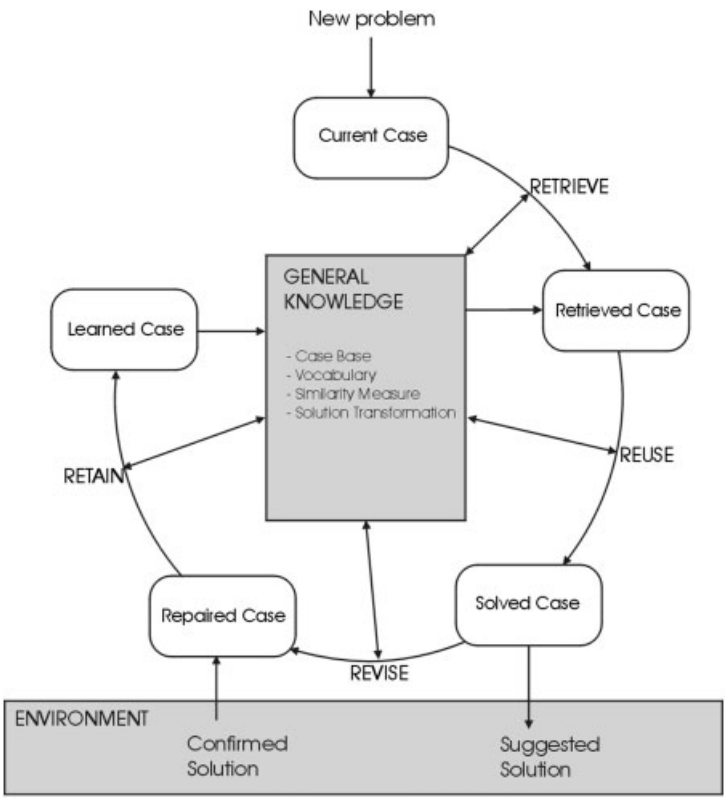

Figure 1. The *CBR-cycle* after Aamodt and Plaza [2].

DOI: 10.1002/spe

better solution to the problem (case adaptation). As the CBR only suggests solutions, there may be a need for a correctness proof or an external validation, which is the task of the phase *revise*. In the *retain* phase, the knowledge learned from this problem is integrated into the system by modifying some knowledge containers. By definition [3], *knowledge container* is the structural element that contains a quantity of knowledge. The idea of the knowledge container is completely different from the traditional module concept in programming. While a module is responsible for a certain subtask, the knowledge container does not complete any subtasks but contains some knowledge relevant to many tasks. Even small tasks require participation of each container. In CBR we identify the following knowledge containers:

(a) used vocabulary;
(b) similarity measure;
(c) case base; and
(d) solution transformation.

The main advantage of this technology is that it can be applied to almost any domain. The CBR system does not try to find rules in the parameters of the problem; it just tries to find similar problems (from the past) and to use the solutions to similar problems as solutions to an actual problem. This approach is therefore extremely suitable for less examined domains—for domains where rules and connections between parameters are not known. Furthermore, in the more examined domains integration of CBR in classical rule-based reasoning systems brings some efficiency. The second very important advantage is that the CBR approach to learning and problem solving is very similar to human cognitive processes—people take into account and use past experience to make future decisions.

3. SIMILARITY BETWEEN CURVES

As we mentioned earlier, our intention was to create a decision support system based on the CBR technology, which deals with time-series data. Since in many real-world domains decisions depend on the behaviour of time diagrams, charts or curves, it would be very useful to implement a system that could help experts in their decision making by analysing curves, comparing them with similar curves from the past and predicting the future behaviour of the current curve on the basis of most similar curves from the past.

The usual way of data representation is a set of points, where a point is an ordered pair (x, y). Very often the pairs are (t, v) where t represents time and v represents a value at time t. Depending on the domain a point can be the result of an experiment, result of a measuring, value of some goods at a given time, etc. When the data are given in this way (as a set of points), it can be represented graphically. When the points are connected they represent a kind of curve [11].

The main problem here, as in almost any CBR system, is to find a good similarity measure. This measure should tell us to what extent two curves are similar. It would be very useful if the result of the similarity measure would be a numerical data.

Looking from another point of view, it is sufficient to compute the distance between two curves. *Distance* is the measurement that can tell to what extent the two curves are different. If the distance $dist(c, c')$ between curves c and c' is known, then the similarity between these curves can

DOI: 10.1002/spe

be computed using the equation

$$sim(c, c') = \frac{1}{1 + dist(c, c')} \tag{2}$$

This equation satisfies the conditions of the similarity definition and the similarity function given in Equation (1). We can tell that the distance is a dual notion for similarity. It follows from this that computing the similarity directly is the same as computing the distance first and then (using this formula) the similarity.

Some authors compute the similarity or distance between curves by a direct composition of distances between points [12–14]. The most frequently used distance measure is the L_2 norm, which is the sum of Euclidian distances between points.

Our approach is different: first, we approximate the curve by an interpolating polynomial through the set of points. Then, we compute the distance between two curves as the distance between their interpolating polynomials. This approach is more flexible and gives the opportunity to compute similarity between sparse and non-equidistant curves, which is not the case with most of the existing approaches.

3.1. Curves and time series

Curve analysis is a popular area in current research, which can be illustrated by a large number of papers published recently. Time-series analysis is by far the most popular subset of curve analysis mainly for the following reason [15]:

> *A random sample of 4000 graphics from 15 of the world's newspapers published from 1974 to 1989 found that more than 75% of all graphics were time series.*

This explosion of interest in time series can be explained by the number of domains in which time series occur: medicine, industry, entertainment, finance, computer science, meteorology and in almost every other field of human activity.

During the research, indicated by different practical needs, many of the task types were separated. Some of the most common are [12,16]:

- Indexing [13,17–19].
- Clustering [20–23].
- Classification [23,24].
- Prediction (forecasting).
- Summarization [25,26].
- Anomaly detection [27–29].
- Segmentation [23,27].

All of the mentioned task types intensively utilize similarity/distance measures between time series. Indexing and clustering make explicit use of distance measures and many approaches to classification, prediction, association detection, summarization and anomaly detection make implicit use of distance measures. Some of the most common similarity measures and techniques for computing similarities are Euclidean distances, dynamic time warping, longest common subsequence similarity, probabilistic methods, general transformations, etc.

DOI: 10.1002/spe

Curve (time-series) databases are generally very large. The suitable choice of representation/approximation is therefore extremely important in curve (time-series) analysis. This has prompted a huge interest in approximate representation of time series. Various solutions that operate mainly with high-level abstraction of the data instead of the original data were developed. Some of the most important are discrete Fourier transform (DFT) [30], discrete wavelet transform (DWT) [18,31,32], singular value decomposition (SVD) [33–35], piecewise linear and piecewise constant models [14,34], adaptive piecewise constant approximation [34], symbolic aggregate approximation [36].

All of the mentioned time-series representations have their advantages and disadvantages. For example, DFT and DWT are often considered as ideal representations for time series, because their first few coefficients contain information about the overall shape of the sequence. However, DFT and DWT are only defined for data whose length is an integer power of two. In contrast, the piecewise constant approximation has exactly the same precision of resolution as the Haar wavelet, but is defined for time series of an arbitrary length.

It is generally impossible to say which of the representations is the best choice. It is more realistic to say that the choice of curve (time-series) representation depends on the domain and the type of application. The choice of the similarity/distance measure depends not only on the domain and type of the application but it depends also on the chosen representation.

In this paper, we present one combination of curve (time-series) representation together with the corresponding similarity/distance measure. We decided to lay stress on generality and robustness for account of efficiency. Our technique is not only able to deal with sparse, uncommon, vague and unusual time series but also with time series whose values (points) are not equidistant. This last mentioned property mostly distinguishes our technique from the techniques mentioned above.

3.2. Classical interpolating polynomial or cubic spline

The main problem in this research was what kind of interpolation should be chosen? If the points are connected just by straight lines then it represents the linear interpolation, but if you want smoother curves then some other kind of interpolation with polynomials must be used. The main choice was between classical interpolating polynomial and cubic spline because of their simplicity and computational efficiency.

The Lagrange interpolating polynomial is the polynomial of degree $n-1$ that passes through n control points $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$. The polynomial can be computed in the following manner [37]:

$$P_j(x) = y_j \prod_{\substack{k=1 \\ k \neq j}}^{x} \frac{x - x_k}{x_j - x_k} \tag{3}$$

where

$$P(x) = \sum_{j=1}^{x} P_j(x) \tag{4}$$

A spline is a piecewise polynomial function that passes through $n+1$ control points (x_0, y_0), $(x_1, y_1), \ldots, (x_n, y_n)$. It can have a locally very simple form, yet it can be globally flexible and smooth at the same time. There are different types of splines. Cubic spline is a spline constructed

DOI: 10.1002/spe

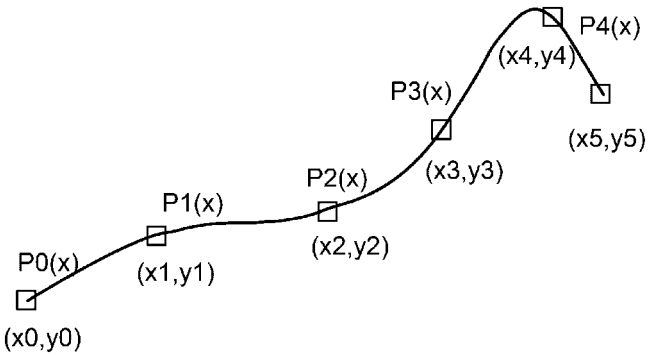

Figure 2. Example of a cubic spline.

of piecewise third-order polynomials such that the function and its first and second derivatives are continuous at the interpolation nodes.

An example of a cubic spline that passes through six points is given in Figure 2. This cubic spline consists of five different third-order polynomials $P0(x), P1(x), \ldots, P4(x)$ that 'connect' six control points.

Generally, a cubic spline that passes through $n+1$ control points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$, consists of n third-order polynomials. If the ith piece of spline is represented in the following manner:

$$y_i(x) = a_i + b_i x + c_i x^2 + d_i x^3 \tag{5}$$

then the coefficients can be computed as follows:

- $a_i = y_i$;
- $b_i = D_i$;
- $c_i = 3(y_{i+1} - y_i) - 2D_i - D_{i+1}$;
- $d_i = 2(y_i - y_{i+1}) + D_i + D_{i+1}$.

The values $D_0, D_1, \ldots, D_n$, which are the first derivates of polynomials in control points, can be computed from the following simple, tridiagonal system:

$$\begin{bmatrix} 2 & 1 & & & & & & \\ 1 & 4 & 1 & & & & & \\ & 1 & 4 & 1 & & & & \\ & & 1 & 4 & 1 & & & \\ \ldots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & & 1 & 4 & 1 & \\ & & & & & 1 & 2 & \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{bmatrix} = \begin{bmatrix} 3(x_1 - x_0) \\ 3(x_2 - x_0) \\ 3(x_3 - x_1) \\ \vdots \\ 3(x_{n-1} - x_{n-3}) \\ 3(x_n - x_{n-2}) \\ 3(x_n - x_{n-1}) \end{bmatrix} \tag{6}$$

DOI: 10.1002/spe

These two kinds of interpolation have the simplest form and the best properties for calculation of the similarities between curves. However, cubic spline was chosen for two main reasons.

- *Power of polynomials*: For $n+1$ points, Lagrange interpolating polynomial has the power n, whereas the power of a cubic spline is always less than or equal to 3. Polynomials with lower power are simpler and more efficient for calculation and memory consumption is less.
- *Oscillation*: The problem of oscillation [38] is shown in Figure 3. In pictures (a) and (b) the Lagrange interpolation polynomial and the cubic spline are shown for the same six points, respectively. In pictures (c) and (d) a seventh 'bad' point is added (the black square), which may be the result of a bad experiment or poor measurement. As it can be seen, interpolating polynomial changes significantly (oscillates) while the cubic spline only changes locally. We found that the local change is more appropriate for the real-world domains.

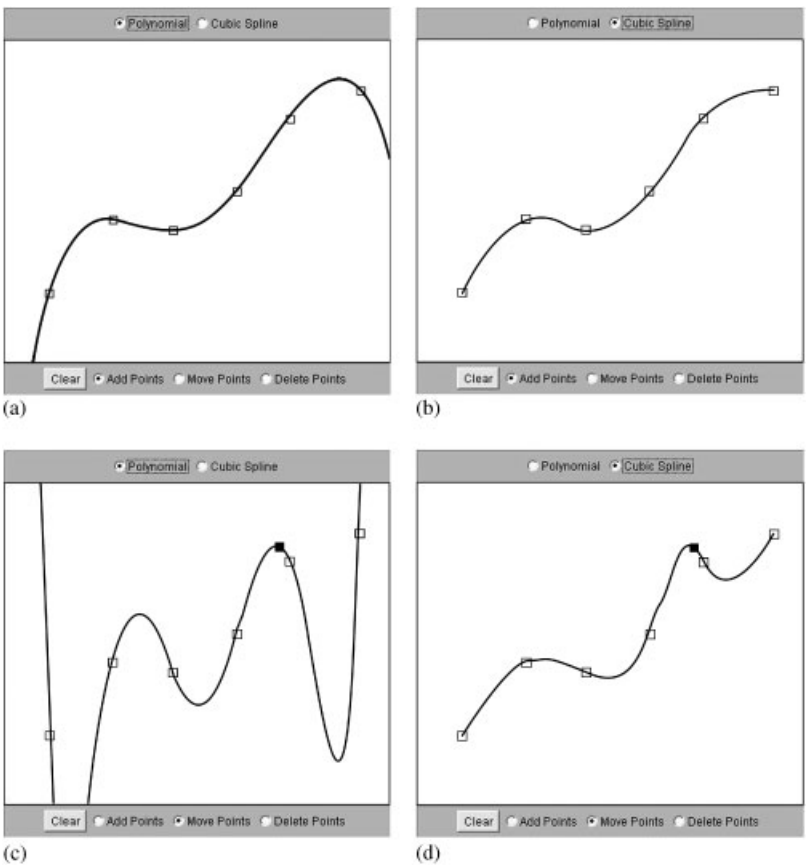

Figure 3. Illustration of oscillation.

DOI: 10.1002/spe

3.3. Computing the distance between curves

As we mentioned earlier, computing the distance (the dual notion for similarity) is sufficient for case-based retrieval. When the curve is given in its analytical form (which is the case with cubic spline) one very intuitive and simple distance measure can be used. The distance between two curves can be represented as an area between these curves over a desired interval as seen in Figure 4. This area can be easily calculated using definite integral. Furthermore, the calculation of definite integral for polynomials is a very simple and efficient operation.

The problem with this approach is the problem of intersections. When two curves have intersection(s) over a given interval, this distance measurement gives wrong results. The characteristic example is given in Figure 5. In the first picture, the curves $f(x)$ and $g(x)$ are shown, whereas in the second picture their difference function $h(x)$ is shown. For the computation of difference between curves f and g in the interval $[a, b]$, the following equation is used:

$$dist(f, g) = \int_a^b f(x)\,\mathrm{d}x - \int_a^b g(x)\,\mathrm{d}x = \int_a^b (f(x) - g(x))\,\mathrm{d}x = \int_a^b h(x)\,\mathrm{d}x \tag{7}$$

The problem here is that this equation gives the result $dist(f, g) = 0$ which, by using Equation (2), gives $sim(f, g) = 1$. That means that these two curves represent the same curve, which is obviously not the case. This undesired property is a consequence of the fact that the surfaces $P1$ and $P2$ on the second picture have the same areas but different signs ($P2 = -P1$). This kind of problem occurs every time when there is an intersection between curves because in that case the areas get subtracted instead of being added.

One solution for this problem could be to calculate intersection points for curves f and g and sum up the absolute values of areas between these points. Calculating intersection points is equivalent to finding zeroes of function $h(x)$. However, computation of zeroes for the polynomial with degree 3 is a very time-consuming operation and requires dealing with complex numbers.

A second solution could be to calculate the square of function $h(x)$. The illustration of this approach can be seen in Figure 6. The main idea here is to use the function $h^2(x)$ instead of the difference between curves (function $h(x)$). The problem with function $h(x)$ is that every time an intersection occurs, the areas $(P1, P2, \ldots)$ change their signs. If we use the function $h^2(x)$ that problem is not actual any more because all surfaces are positive $(P1', P2', \ldots)$. The most important

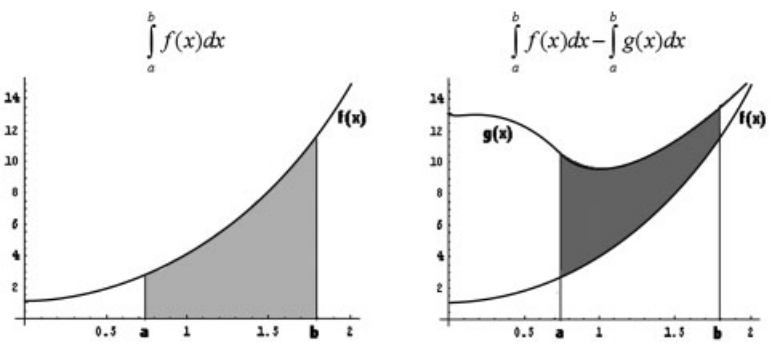

Figure 4. Definite integral and surface between curves.

DOI: 10.1002/spe

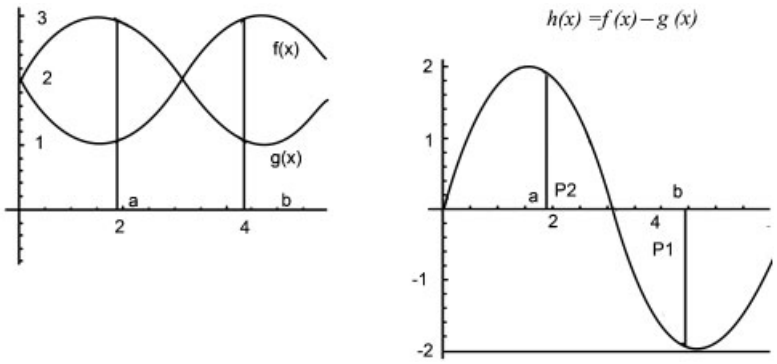

Figure 5. Problem with intersection(s).

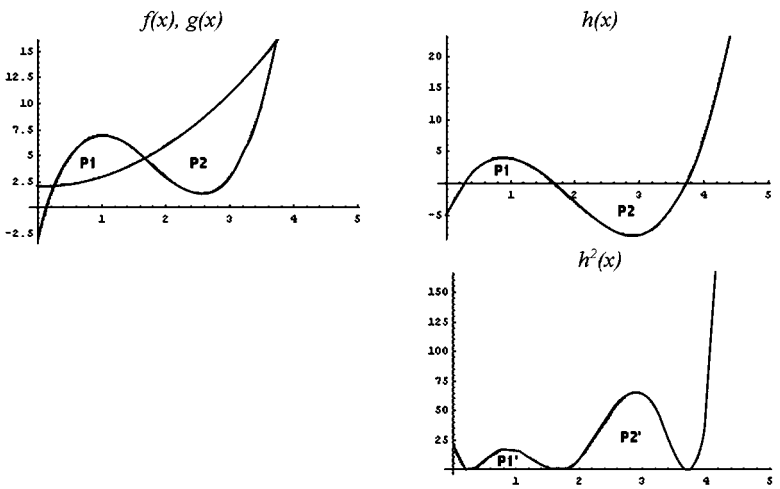

Figure 6. Calculation of $h^2(x)$ function.

property here is that these surfaces are proportional. It means that if $P1 < P2$ for the function $h(x)$ then $P1' < P2'$ for the function $h^2(x)$. This is important because the order of similar curves is preserved by this transformation.

When the function $h^2(x)$ is calculated, the distance between two curves can be calculated by simply summing up all the areas $(P1', P2', \ldots)$ using the definite integral.

4. CuBaGe—CURVE BASE GENERATOR

The system with the characteristics described in the previous section was implemented in a Java environment (JDK 1.3). The polynomials are represented as arrays of their coefficients. Each curve is represented as a set of its points and a set of polynomials between the points by the definition of cubic spline. The splines are calculated using the procedure described in Section 3.2. The distance

DOI: 10.1002/spe

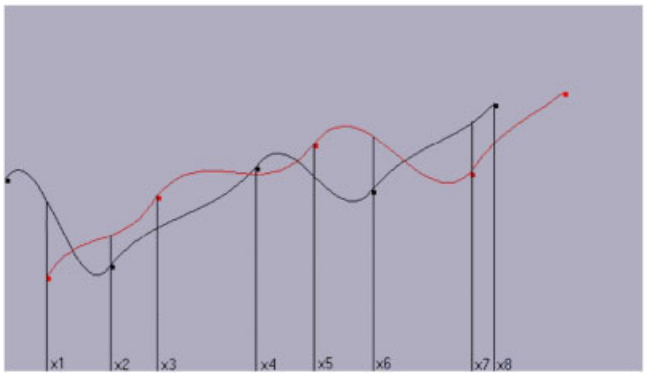

Figure 7. Computing distance between two curves.

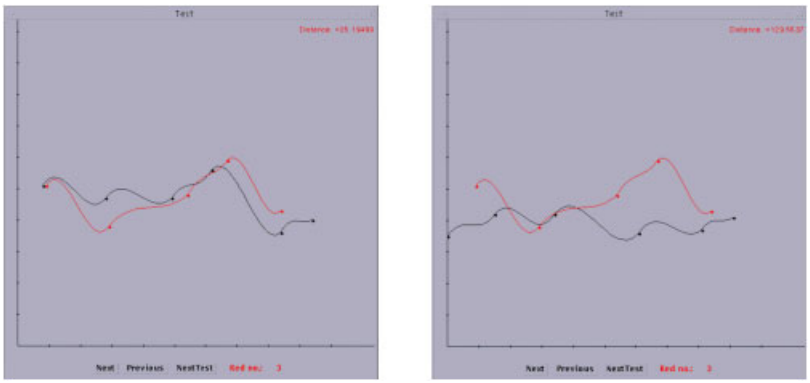

Figure 8. Most similar and 25th similar curve.

between two curves is calculated using the definite integral of the function $h^2(x)$ as described in the previous section. However, the distance must be partially computed between each two points of both curves because the polynomials of the splines are changed in each point (Figure 7). To compute the global distance between two curves, all these partial distances simply have to be summed.

For a testing purpose, a randomly generated database of curves was created. The database consists of 1200 curves. Each curve has a set of points. The number of points is also a random number (more than 4 points and less than 25). Initially, the system reads those sets of points (from one input file), creates splines for every curve and internally stores the curves, which represents the case base or, rather, a curve base. Then, in the same way, the system reads a curve from another input file, and that curve represents a current problem that has to be solved. After that the system retrieves from the curve base the curves that are most similar to the problem curve, and shows the retrieved curves sorted by distance.

Temporarily, a graphical user interface was created so that the results of the system can be seen interactively. In Figure 8, some snapshots from the system can be seen. For the same problem curve, represented by the lighter curve, in the left picture the most similar curve from the curve base is shown (distance = 2619499). In the right picture the 25th curve (sorted by distance) is shown

DOI: 10.1002/spe

(distance = 1 296 637). As can be seen, the system really retrieves the most (intuitively) similar curve from the curve base.

5. REAL-WORLD APPLICATION—FINANCIAL PREDICTION

As an application domain, the predicting payment process in the company 'Novi Sad Fair' was chosen [9,39]. The management of 'Novi Sad Fair' wanted to know how high would be the payment of some services over time, with respect to its invoicing. They provided the data about payment and invoicing process for the last 3 years for each exhibition. Every year there are between 25 and 30 exhibitions; hence, the database consisted of around 80 payment and invoicing processes. These processes were represented by a set of points, where each point was given with the time of measuring (in days since the beginning of the process) and the value of payment or invoicing on that day. It follows from this that these processes could be represented as curves.

Measuring of the values of payment and invoicing was done every 4 days from the beginning of the invoice process for a duration of 400 days, which meant that the curve consisted of 100 points. By analysing these curves, we observed that the process of invoicing usually started several months before the exhibition and that the value of invoicing rapidly grew approximately until the date of exhibition. After that date, the value of invoicing remained approximately the same until the end of the process. The moment when the value of invoicing reaches some constant values and stays the same till the end of the process is called the *time of saturation* for the invoicing process and the corresponding value the *value of saturation*.

The process of payment starts several days after the corresponding process of invoicing (for the same exhibition). After that the value of payment grows, but not as rapidly as the value of invoicing. At the moment of exhibition, the value of payment is between 30 and 50% of the value of invoicing. After that the value of payment continues to grow until the moment when it reaches a constant value and stays approximately constant until the end of the process. That moment we call *time of saturation* for the payment process, and the corresponding value the *value of saturation.*

The time of payment saturation usually comes few months after the time of invoice saturation, and the payment value of saturation is always less than or equal to the invoice value of saturation. Our analysis shows that the payment value of saturation is between 80 and 100% of the invoice value of saturation. One characteristic invoice and a corresponding payment curve are shown in Figure 9 as 'old payment curve' and 'old invoice curve'. The points of saturation (time and value) are represented by the larger points on the curves.

The system works as follows: it first reads the input data from two databases: one database contains the information about all invoice processes for every exhibition in the past 3 years, whereas the other database contains the information about the corresponding payment processes. Each process (invoice or payment) is represented by a set of points; the process can be considered as a curve. After that the system creates splines for each curve (invoice and payment) and internally stores the curves in some kind of a list, where an element of the list consists of an invoice curve and the corresponding payment curve.

In the same way the system reads the problem curves from another database. The problem for us is the invoice curve and its corresponding problem curve at the moment of exhibition. At that moment the invoice curve reaches its saturation point while the payment curve is still far away from

DOI: 10.1002/spe

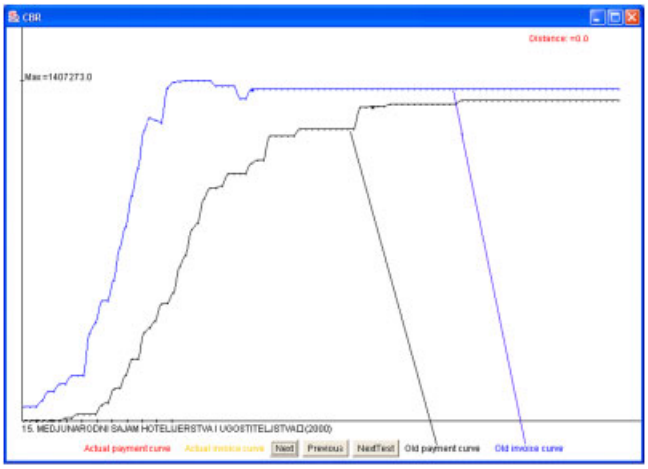

Figure 9. Payment and invoice curve.

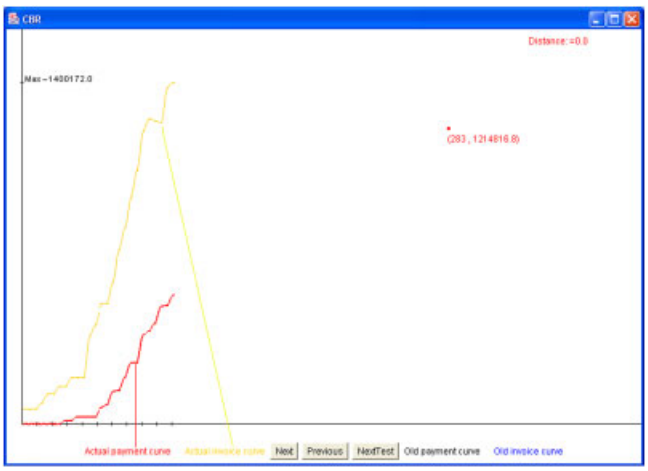

Figure 10. Problem payment and invoice curve.

its saturation point. These curves are shown in Figure 10 as 'actual payment curve' and 'actual invoice curve'.

The solution to this problem would be finding the saturation point for the payment curve. This means that the system helps experts by suggesting and predicting the time and value at which the payment process will reach its saturation point.

The saturations point is calculated using 10% of the most similar payment curves from the database of previous payment processes. The distance is calculated using the previously described algorithm. Since the values of saturation are different for each exhibition, each curve from the

DOI: 10.1002/spe

database must be scaled with a factor so that the invoice values of saturation of the old curve and the actual one would be the same. That factor is easily calculated as

$$Factor = \frac{actual\_value\_of\_saturation}{old\_value\_of\_saturation} \tag{8}$$

where the actual value of saturation is in fact the value of the invoice at the time of exhibition.

The final solution is then calculated by using the payment saturation points of the 10% of the most similar payment curves. Saturation points of the similar curves are multiplied with a kind of 'goodness' and then summed. The values of goodness are directly proportional to the similarity between the old and actual curves, but the sum of all goodnesses must be 1. Since the system calculates the distance, similarity is calculated as shown in Equation (2).

The goodness for each old payment curve is calculated as

$$goodness_i = \frac{sim_i}{\sum_{all\_j} sim_j} \tag{9}$$

At the end, the final solution—payment saturation point is calculated as

$$sat\_point = \sum_{all\_i} goodness_i \cdot sat\_point_i \tag{10}$$

The system draws the solution point in the diagram together with the saturation time and value. Solution point can be seen in Figure 10 as a point with its coordinates. After that similar curves sorted by similarity can be seen along with the distance between them (in the upper right corner). Both the actual payment and invoice curve as well as a similar old payment and an invoice curve can be seen in Figure 11.

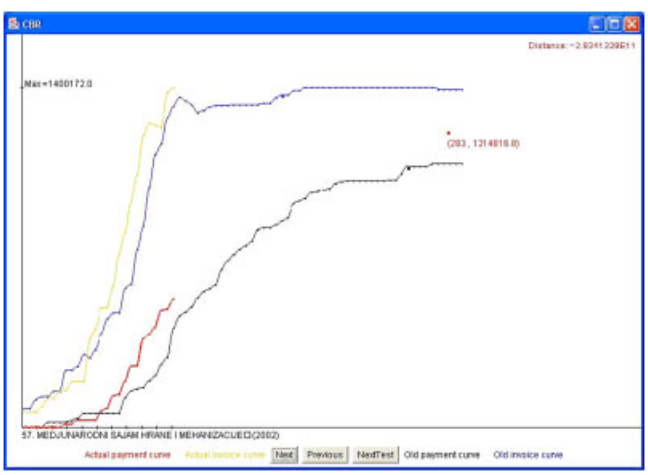

Figure 11. Problem payment and invoice curve and some similar curves from the database.

DOI: 10.1002/spe

5.1. Experiments

We conducted a set of experiments using the previously described methodology. The intention was to prove that the described method can manage non-standard time series equally well as the standard ones. Those non-standard series include sparse time series (series that have long time intervals without known values) and non-equidistant time series (series in which points are not equally distributed along the time axis).

As a good example, the process of calculating the saturation point for curve number 2 is given in Table I. Saturation point is calculated using 10 most similar curves. In the first column, the record number of the most similar curves is given. Saturation points of the corresponding similar curves are given in the column 'saturation point'. The values of these saturation points must be scaled by an appropriate value so that they could be used for calculation. Scale value is calculated using Equation (8) and is given in the column 'Scale'. Normalized saturation points are then used for calculation. Similarity between the actual curve (record number 2) and a corresponding curve is given in the 'similarity' column. Goodness represents the contribution of the corresponding curve to the calculation of the actual saturation point. It is directly proportional to the similarity value and is calculated using Equation (9).

The final solution (saturation point of the actual curve) is then calculated as a linear combination of normalized saturation points of these curves. The coefficients for linear combination are values of goodness given in Equation (10). In Table I the real saturation point and the error made by our system are also given.

As the next step, we conducted a K-fold cross-validation (where $K = 10$) to evaluate the results of the *CuBaGe* system. The original database was partitioned into 10 subsets of equal size. One of the 10 subsets was retained as the validation data for testing the system, and the remaining nine subsets were used as training data. The cross-validation algorithm was then repeated 10 times, with each of the 10 subsamples used exactly once as the validation data. The errors that were calculated (made in time dimension and value dimension) are shown in Table I. All results (errors) from the iterations were then averaged to produce a single estimation.

Table I. Computing the solution for curve number 2.

Curve no.	Saturation point		Scale	Normalized saturation point		Similarity	Goodness
26	281	1016239	4.4210415	281	4492835.00	7.41264E−14	0.270515654
43	149	3697192	1.2623929	149	4667309.00	4.49298E−14	0.163966103
68	177	619940	7.7333126	177	4794190.00	3.1515E−14	0.115010156
46	357	4031816	1.0855181	357	4376609.50	2.79526E−14	0.102009587
33	333	1102806	3.4476216	333	3802057.80	2.19667E−14	0.080164732
31	221	1416572	3.2382340	221	4587191.50	1.6432E−14	0.059966748
62	185	2850388	1.5151005	185	4318624.00	1.49321E−14	0.054492899
17	321	1067056	4.4963370	321	4797843.50	1.47348E−14	0.053772887
75	213	1970956	2.2591212	213	4452628.50	1.38645E−14	0.050596884
39	353	3433443	1.3006786	353	4465806.00	1.35651E−14	0.049504354
Computed saturation point				252.762	4498063.50		
Real saturation point				265	4425584.00		
Error (%)				3.06	1.64		

DOI: 10.1002/spe

In Table II some of the results of the 10-fold cross-validation algorithm are given. The real saturation points and the saturation points calculated by the *CuBaGe* system are given in the table. In the column 'Δ' the variance between real and computed saturation point is shown. In the next column, the percentage of error for a particular curve is shown. The average of all errors for time and value dimension is given in the last column.

It is obvious that the error in time dimension is higher than on value dimension. This is completely understandable from the operational manager's point of view, because the time of saturation depends on many other economical and political reasons. This difference can be confirmed by a large variation of saturation time in the existing database of curves (a part of it can be seen in Table I—'saturation point'). A similar trend may be observed in the following experiments as well.

Table II. Part of the 10-fold cross-validation.

Curve no.	Real saturation point		Computed saturation point		Δ		Percent of error (%)	
1	289	576998	228.88	602278.75	−60.12	25280.75	15.03	4.38
2	265	4425584	252.762	4498063.5	−12.238	72479.5	3.06	1.64
3	293	640952	234.418	674823.063	−58.582	33871.06	14.65	5.28
4	353	61662912	256.503	61871660	−96.497	208748	24.12	0.34
5	325	11038734	251.434	11018625	−73.566	−20109	18.39	0.18
6	277	666613	252.498	686351.813	−24.502	19738.81	6.13	2.96
7	309	375608	216.822	372590.375	−92.178	−3017.63	23.05	0.80
8	241	355064	294.821	440066	53.821	85002	13.46	23.94
9	353	2929711	300.843	2939542	−52.157	9831	13.04	0.34
				...				
				...				
Global error							14.11699	5.879581

Table III. Fixed number of removed points.

Experiment	Time error (%)	Value error (%)
F01	14.11699	5.8795805
F02	13.040847	6.779536
F03	15.299633	6.594407
F04	14.28599	6.1562805
F05	14.445005	6.301309
F06	13.462544	6.403636
F07	15.202986	6.435486
F08	14.777561	6.88809
F09	14.017353	6.692904
F10	14.213959	6.992091
F11	13.713492	6.79082
F12	16.24996	7.730693
F13	14.553808	7.69725
F14	16.16749	7.011806
F15	15.02239	7.798069
F16	16.465824	7.657989
F17	12.756111	7.056797

DOI: 10.1002/spe

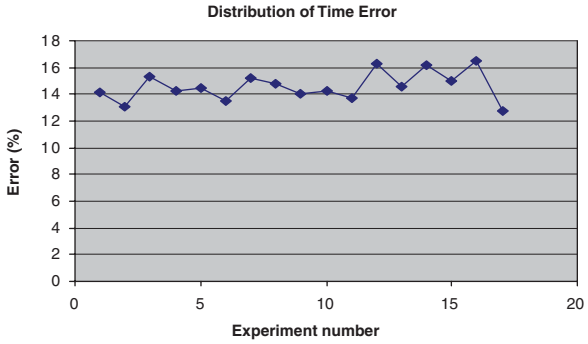

Figure 12. Distribution of time error (fixed).

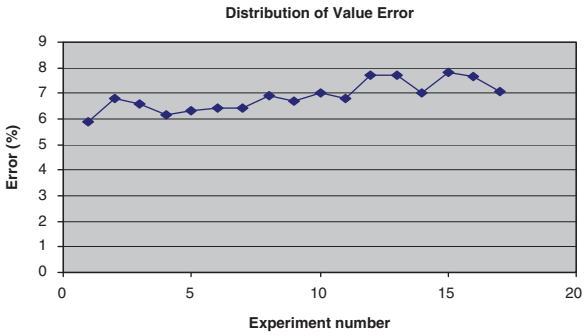

Figure 13. Distribution of value error (fixed).

In the next phase, we conducted 17 experiments that illustrate the ability of the *CuBaGe* system to deal with sparse curves. Here, some points were removed following this rule: in the ith experiment, only the points with indexes that are multiples of i are kept. For example, in the 12th experiment only points with these indexes are kept: 0, 12, 24, 36, 48, 60, 72, 84, ..., whereas in the first experiment none of the points were removed, and it was equivalent to the previously described experiment. This discarding of points was done in all curves, both in validating and in the test data. On this kind of data, the 10-fold cross-validation algorithm was executed.

We stopped the experiments at number 17 because with higher discarding index (more than 17), some of the validation curves that had to be truncated within the time of the invoice saturation would consist of only one point. This is unusable for any kind of prediction, and furthermore it is impossible to compute spline through a single point.

The results of these experiments are shown in Table III. In the table, the global errors of each 10-fold cross-validation experiment are shown. In Figures 12 and 13, the distribution of time and value errors in relation to the number of discarded points is shown. It is obvious that the accuracy of the system does not decrease as the number of removed point rises.

In the last phase, we further conducted 17 experiments with the intention to illustrate the ability of the *CuBaGe* system to deal with non-equidistant curves. The idea of these experiments is similar

DOI: 10.1002/spe

to the previous one. In the ith experiment, the first point is kept, and after it a random number (from the interval $[0, i]$) of points are removed. Accordingly, the next point is kept, and again after it a random number (from the interval $[0, i]$) of points are removed, etc. For example, the indexes of points that are kept in the 12th experiment could be 0, 3, 14, 17, 25, 28, 34, 41, 52, 55, ... Again, in the first experiment none of the points were removed.

In Table IV, the results of these experiments are given, whereas in Figures 14 and 15 the distributions of time and value errors are given. Again, there is no significant decrease in the accuracy of the system as the number of removed points rises, even in the situation where the number and distribution of points are random.

The presented experiments demonstrate generality and permanence of the *CuBaGe* system in this domain. The system gave the same results with standard time series (where points are equidistant and relatively 'bushy'), as with sparse and non-equidistant time series where a big amount of points were removed (original curves consisted of 400 points, and in experiment F17 each curve consists

Table IV. Random number of removed points.

Experiment	Time error (%)	Value error (%)
R01	16.11699	5.8795805
R02	13.887385	6.7979
R03	16.783285	7.35782
R04	15.951675	6.951407
R05	15.294256	6.252278
R06	15.85167	6.126275
R07	14.489132	6.649499
R08	16.001661	6.542799
R09	13.9076	8.251835
R10	14.316044	7.298194
R11	13.545891	6.926044
R12	15.298765	6.396808
R13	14.793287	7.727208
R14	14.64713	6.776858
R15	16.55061	6.721552
R16	15.571243	6.399883
R17	14.334753	7.097614

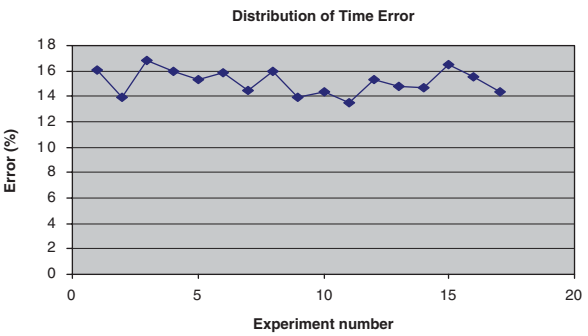

Figure 14. Distribution of time error (random).

DOI: 10.1002/spe

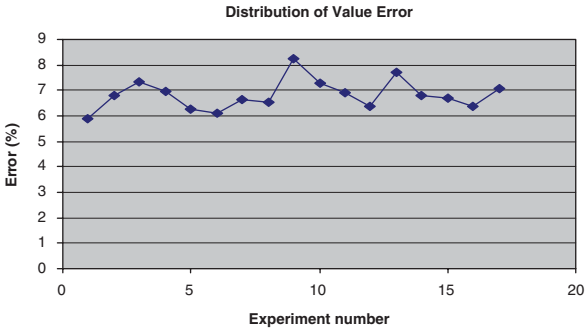

Figure 15. Distribution of value error (random).

of only 22 points). All these facts indicate that the system could be applied in other domains, where the data are represented in a similar manner, with a high level of precision and reliability.

The proposed mathematical concept for curve representation (spline) is very robust and flexible. Robustness and flexibility are the main properties of splines, which have the following further advantages: simplicity of construction, accuracy of evaluation, capacity to approximate complex shapes through curve fitting, etc.

Furthermore, the proposed distance/similarity measure that uses definite integrals is the most adequate for spline representation, and guarantees a high level of accuracy and correctness. The computation of definite integrals is efficient, and this surface-based distance measure is very intuitive and simple.

The presented approach utilizes a well-known data mining classification technique, the k-nearest neighbour (kNN). In fact, this approach does not calculate the class of retrieved time series; it only retrieves k most similar time series, according to the explained distance measure, and uses them for the prediction. In this stage, the kNN algorithm gives promising results, but it would be interesting to test some other classification techniques in the future.

However, one of the greatest contributions of this system could be its ability to keep the same accuracy even with curves whose points are not equidistant. When we removed a random number of points in a random order, the system did not show any important oscillations. This property mostly distinguishes our system from the standard approaches in the curve/time-series analysis (DFT, DWT, SVD, etc.).

6. RELATED WORK

Although CBR is successfully used in many areas (aircraft conflict resolution in air traffic control [40], optimizing rail transport [41], subway maintenance, optimal job search, support to help-desks, intelligent search on internet) [3], it is not very often used in combination with data warehousing and in collaboration with classical online analytical processing, probably due to novelty of this technique. CBR does not require causal model or deep understanding of a domain, and therefore it can be used in domains that are poorly defined, where information is incomplete, contradictory or where it is difficult to get sufficient domain knowledge [2,3,42]. All this is typical for business processing.

DOI: 10.1002/spe

Besides CBR, other possibilities are in rule base knowledge or knowledge discovery in a database, where knowledge evaluation is based on rules [2]. The rules are usually generated by combining propositions. As the complexity of the knowledge base increases, maintaining becomes problematical, because changing rules often imply a lot of reorganization in the rule base system. On the other hand, it is easier to add or delete a case in a CBR system, which finally provides the advantages in terms of learning and explaining ability.

The main idea of our research was to develop a robust and a general architecture for curve representation and for indexing time-series databases. Also, for a given kind of curve representation, the corresponding similarity measure was modelled. The presented architecture can be employed not only equally well in conventional time series (where all values are known) but also in some non-standard time series (sparse, vague, non-equidistant). Dealing with non-standard time series is the biggest advantage of the presented architecture.

The proposed method is generally not applicable in indexing and manipulating large databases of time series. Most of the actual approaches utilize restricted dimensionality reduction techniques for optimizing searches in databases [12,13,17,30–33]. Also, these algorithms (DFT, DWT, SVD, etc.) do not return the exact matches or the most similar time series from the database. Instead, they return only candidates that have to be investigated using more sophisticated methods. Our system could be successfully applied in these situations.

One interesting application of this system could be in combination with DFT or DWT. These approaches are most commonly used in time-series indexing. Unfortunately, their disadvantage is that they are only defined for data where the number of points is an integer power of 2 [12,30,32]. Since, our system has the ability to 'fill the gap' between the points (which is a natural result of using splines), it can be used to create a curve from the data whose length is not an integer power of two. After that, since the system creates continuous and smooth function, it would be a trivial task to create a set of equidistant points with a length equal to integer power of two (for an arbitrary power). Afterwards, any of the algorithms developed for DFT and DWT could be applied.

Even if the proposed approach with the kNN classification algorithm for similar time-series selection gives high-quality results, it would be interesting to test the system with some other classification algorithms. Furthermore, testing some other prediction techniques based on regression or extrapolation seems motivating. Moreover, it would be interesting to compare different combinations of classification/prediction algorithms and their results in different domains.

Although the prediction of curves has many possibly interesting applications (in telecommunications [42], medicine [43,44], economics [45]), the approach with the CBR technique is rarely used [46,47]. This approach [9,10,39] based on looking for similarities in curves and predicting future trends is interesting and seems to have a potential for application in different domains in future research.

In situations where the points in time series are non-equidistant (reasons can be numerous: unknown data, error in measurement, malfunction in measurement device, etc.), our system is supposed to give promising results, based on the conducted experiments.

7. CONCLUSION

The presented approach of combining CBR systems with time-series data is a promising area of research. The application of the proposed *CuBaGe* system in payment prediction process has

DOI: 10.1002/spe

numerous advantages. For example, an operational manager could make important business decisions based on the CBR predictions and take appropriate actions: make payment delays shorter, make the total of payment amount higher, secure payment guarantee on time, reduce the risk of payment cancellation and inform senior managers on time. Senior managers could use this information to plan better possible investments and new exhibitions. This would be based on the amount of funds and the time of their availability as predicted by the CBR system.

By combining graphical representation of predicted values with the most similar curves from the past, the system enables better and more focused understanding of predictions with respect to real data from the past.

The presented system is not only limited to this case study it can also be applied to other business values as well (expenses, investments, profit). Furthermore, the system can be used in other non-business domains, where some decisions depend on the curves, behaviour (medicine, physics, etc.). The good properties of the system (such as generality, robustness, ability to deal with sparse and non-equidistant time series, etc.) open a wide range of possible application domains.

Further possible research can be directed towards using some other interpolation functions, which will depend on the implementation domain. Linear interpolation can be realized very easily, but it would be interesting to test some other interpolation functions. Also, it would be very interesting to generalize the system so that it could work not only with just two-dimensional curves but also with three-dimensional curves and surfaces and in higher dimensions. This would be applicable in domains where decisions depend on more than two factors.

REFERENCES

1. Budimac Z, Kurbalija V. Case-based reasoning—A short overview. *Conference of Informatics and IT*, Bitola, 2001.
2. Aamodt A, Plaza E. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications* 1994; **7**:39–59.
3. Lenz M, Bartsh-Sporl B, Burkhard H-D, Wess S, Goos G, Van Leeuwen J, Bartsh B. *Case-based Reasoning Technology*: *From Foundations to Applications*. Springer: Berlin, 1998.
4. Kurbalija V, Ivanović M, Budimac Z. CBR—Case-based reasoning. *Journal of Information Technology and Multimedia Systems* 2003; **infoM 5**:14–20.
5. Kolodner J. *Case-based Reasoning*. Morgan Kaufmann: Los Altos, CA, 1993.
6. Kurbalija V, Ivanović M. CaseBaseGenerator for intelligent systems. *Novi Sad Journal of Mathematics* 2005; **35**(1):25–40.
7. Ivanović M, Kurbalija V, Budimac Z, Semnic M. Role of case-based reasoning in neurology decision support. *Proceedings of the Fifth Joint Conference on Knowledge-based Software Engineering* '*JCKBSE 2002*', Maribor, Slovenia, 11–13 September 2002; 255–264.
8. Kurbalija V, Ivanović M. CBG—A framework for intelligent decision support systems. *Proceedings of the Fourth International Conference on Informatics and Information Technology* '*Molika 2003*', Bitola, Macedonia, 11–14 December 2003; 118–128.
9. Simić D, Kurbalija V, Budimac Z. Improving the CBR system for financial predictions. *Proceedings of the International Conference on Information and Knowledge Engineering* '*IKE 04*', Las Vegas, NV, U.S.A., 21–24 June 2004; 561–567.
10. Simić D, Kurbalija V, Budimac Z. An application of case-based reasoning in multidimensional database architecture. *Proceedings of the 5th International Conference on Data Warehousing and Knowledge Discovery* (*DaWaK*) (*Lecture Notes in Computer Science*, vol. 2737). Springer: Berlin, Heidelberg, New York, 2003; 66–75.
11. Kurbalija V. *On Similarity of Curves—Project Report*. AI Lab, Humboldt University: Berlin, 2003.
12. Ratanamahatana CA, Lin J, Gunopulos D, Keogh E, Vlachos M, Das G. Mining time series data. *Data Mining and Knowledge Discovery Handbook*, Maimon O, Rokach L (eds.). Springer: Berlin, 2005. ISBN: 978-0-387-24435-8.
13. Faloutsos C, Ranganathan M, Manolopoulos Y. Fast subsequence matching in time-series databases. *Proceedings of the ACM SIGMOD Int'l Conference on Management of Data*, Minneapolis, MN, 25–27 May 1994; 419–429.
14. Yi B, Faloutsos C. Fast time sequence indexing for arbitrary Ip norms. *Proceedings of the 26th Int'l Conference on Very Large Databases*, Cairo, Egypt, 10–14 September 2000; 385–394.
15. Tufte E. *The Visual Display of Quantitative Information*. Graphics Press: Cheshire, CT, 1983.

DOI: 10.1002/spe

16. Das G, Gunopulos D, Mannila H. Finding similar time series. *Principles of Data Mining and Knowledge Discovery* (*Lecture Notes in Artificial Intelligence*, vol. 1263). Springer: Berlin, 1997.
17. Chakrabarti K, Keogh E, Pazzani M, Mehrotra S. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM Transactions on Database Systems* 2002; **27**(2):188–228.
18. Kahveci T, Singh A. Variable length queries for time series data. *Proceedings of the 17th Int'l Conference on Data Engineering*, Heidelberg, Germany, 2–6 April 2001; 273–282.
19. Popivanov I, Miller RJ. Similarity search over time series data using wavelets. *Proceedings of the 18th Int'l Conference on Data Engineering*, San Jose, CA, 26 February–1 March 2002; 212–221.
20. Aach J, Church G. Aligning gene expression time series with time warping algorithms. *Bioinformatics* 2001; **17**:495–508.
21. Debregeas A, Hebrail G. Interactive interpretation of Kohonen maps applied to curves. *Proceedings of the 4th Int'l Conference of Knowledge Discovery and Data Mining*, New York, NY, 27–31 August 1998; 179–183.
22. Kalpakis K, Gada D, Puttagunta V. Distance measures for effective clustering of ARIMA time-series. *Proceedings of the IEEE Int'l Conference on Data Mining*, San Jose, CA, 29 November–2 December 2001; 273–280.
23. Keogh E, Pazzani M. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. *Proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining*, New York, NY, 27–31 August 1998; 239–241.
24. Geurts P. Pattern extraction for time series classification. *Proceedings of Principles of Data Mining and Knowledge Discovery*, *5th European Conference*, Freiburg, Germany, 3–5 September 2001; 115–127.
25. Indyk P, Koudas N, Muthukrishnan S. Identifying representative trends in massive time series data sets using sketches. *Proceedings of the 26th Int'l Conference on Very Large Data Bases*, Cairo, Egypt, 10–14 September 2000; 363–372.
26. van Wijk JJ, van Selow ER. Cluster and calendar-based visualization of time series data. *Proceedings of IEEE Symposium on Information Visualization*, 25–26 October 1999. IEEE Computer Society: Silver Spring, MD, 1999; 4–9.
27. Guralnik V, Srivastava J. Event detection from time series data. *Proceedings of the 5th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining*, San Diego, CA, 15–18 August 1999; 33–42.
28. Keogh E, Lonardi S, Chiu W. Finding surprising patterns in a time series database in linear time and space. *The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alb., Canada, 23–26 July 2002; 550–556.
29. Shahabi C, Tian X, Zhao W. TSA-tree: A wavelet based approach to improve the efficiency of multi-level surprise and trend queries. *Proceedings of the 12th Int'l Conference on Scientific and Statistical Database Management*, Berlin, Germany, 26–28 July 2000; 55–68.
30. Agrawal R, Faloutsos C, Swami AN. Efficient similarity search in sequence databases. *Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms*, Chicago, IL, U.S.A. (*Lecture Notes in Computer Science*, vol. 730), Lomet DB (ed.). Springer: London, 13–15 October 1993; 69–84.
31. Chan K, Fu AW. Efficient time series matching by wavelets. *Proceedings of the 15th IEEE International Conference on Data Engineering*, Sydney, Australia, 23–26 March 1999; 126–133.
32. Wu Y, Agrawal D, El Abbadi A. A comparison of DFT and DWT based similarity search in time-series databases. *Proceedings of the 9th ACM CIKM Int'l Conference on Information and Knowledge Management*, McLean, VA, 6–11 November 2000; 488–495.
33. Kanth KV, Agrawal D, Singh A. Dimensionality reduction for similarity searching in dynamic databases. *Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, Seattle, WA, U.S.A., 1–4 June 1998; 166–176.
34. Keogh E, Chakrabarti K, Mehrotra S, Pazzani M. Locally adaptive dimensionality reduction for indexing large time series databases. *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Santa Barbara, CA, U.S.A., 21–24 May 2001; 151–162.
35. Kom F, Jagadish H, Faloutsos C. Efficiently supporting ad hoc queries in large data sets of time sequences. *Proceedings of the SIGMOD International Conferences*, Tucson, AZ, 1997; 289–300.
36. Lin J, Keogh E, Lonardi S, Chiu B. A symbolic representation of time series, with implications for streaming algorithms. *Workshop on Research Issues in Data Mining and Knowledge Discovery*, *Eighth ACM SIGMOD*, San Diego, CA, 13 June 2003.
37. Wolfram Research. http://mathworld.wolfram.com/ [January 2003].
38. Tobias von Petersdorf Home Page. http://www.wam.umd.edu/∼petersd/, Department of Mathematics, University of Maryland [March 2004].
39. Simić D, Budimac Z, Kurbalija V, Ivanović M. Case-based reasoning for financial prediction. *Lecture Notes in Artificial Intelligence* 2005; **3533**:839–842. ISSN: 0302-9743 (*Proceedings of the 18th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, *IEA/AIE 2005*, Bari, Italy, 22–24 June 2005).
40. Chiu C, Chiu N-H, Hsu C-I. Intelligent aircraft maintenance support system using genetic algorithms and case-based reasoning. *The International Journal of Advanced Manufacturing Technology* 2004; **24**(5):440–446. DOI: 10.1007/s00170-003-1707-x.
41. Cui Y, Tang Z, Dai H. Case-based reasoning and rule-based reasoning for railway incidents prevention. *Proceedings of ICSSSM '05. 2005 International Conference on Services Systems and Services Management*, vol. 2, Chongqing, China, 13–15 June 2005; 1057–1060.

42. Yan L, Wolniewicz RH, Dodier R. Predicting customer behavior in telecommunications. *IEEE Intelligent Systems* 2004; **19**(2):50–58.
43. Davies HE, Court JB, Burn C. Prediction of the initial slope of the acute clonogenic survival curve by the post-irradiation behaviour of cytochalasin-induced polykaryons. *International Journal of Radiation Biology* 1995; **68**(6):631–645.
44. Dias MS, Takeda MN, Koskinas MF. Application of Monte Carlo simulation to the prediction of extrapolation curves in the coincidence technique. *Applied Radiation and Isotopes* 2006; **64**(10–11):1186–1192.
45. Bernadell C, Coche J, Nyholm K. Yield curve prediction for the strategic investor. *European Central Bank Working Paper Series No. 472*, 2005.
46. Tang Y. Time series extrapolation using hierarchical case-based reasoning. *Proceedings of the 24th IASTED International Conference on Signal Processing*, *Pattern Recognition*, *and Applications*, Innsbruck, Austria, 15–17 February 2006, Hamza MH (ed.). International Association of Science and Technology for Development. ACTA Press: Anaheim, CA, 2006; 194–199.
47. Nakhaeizadeh G. Learning prediction of time series—A theoretical and empirical comparison of CBR with some other approaches. *Selected Papers From the First European Workshop on Topics in Case-based Reasoning*, 1–5 November 1993 (*Lecture Notes in Computer Science*, vol. 837). Wess S, Althoff K, Richter MM (eds.). Springer: London, 1994; 65–76.

DOI: 10.1002/spe