UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA MATEMATIKU I INFORMATIKU

Vladimir Kurbalija

TIME SERIES ANALYSIS AND PREDICTION USING CASE BASED REASONING TECHNOLOGY

~ Ph.D. Thesis ~

Mentor:
Mirjana Ivanović, Ph.D.

Novi Sad, 2009.

PREFACE

This thesis describes one promising approach where a problem of time series analysis and prediction was solved by using Case Based Reasoning (CBR) technology. Case-Based Reasoning has become successful technique for knowledge-based systems in different domains. This promising technique is based on the use of previous experience in a form of cases so as to understand better and solve new problems in a particular domain. The main assumption in CBR is that, for many particular domains, similar problems usually have similar solutions.

Time series analysis covers many different task types including classification, indexing, clustering, prediction etc., which deals with the data represented as a sequence of data points, measured at successive time periods. For a long time, time series analysis was based solely on statistical methods. Recently, some of the research was done in the area of applying different Machine Learning and Data Mining techniques in time series analysis. However, there are only a few papers on applying the CBR in time series analysis.

This Thesis consists of the following seven chapters:

1. Introduction
2. Foundations of CBR
3. Time-Series Analysis
4. Time-Series Forecasting
5. *CuBaGe* System
6. *CuBaGe* in Financial Forecasting
7. Conclusion

In the first Chapter an overview of this field is given and the goals of the Thesis are defined. Chapter two describes foundations of CBR technology together with all necessary definitions. Known techniques in time series analysis and time series prediction are given in chapters three and four respectively. The system for time series analysis and forecasting based on the CBR technology *CuBaGe* (Curve Base Generator) is presented in the fifth Chapter. One specific application of the *CuBaGe* system in a financial domain, together with results of experiments is given in the sixth Chapter. Chapter seven concludes the Thesis and lists some possibilities for further work.

At this moment, I would like to thank warmly to all the members of the committee for their patience and valuable suggestions regarding this thesis. But, my special thanks and gratitude are addressed to prof. dr. Mirjana Ivanović and prof. dr. Zoran Budimac who patiently guided me through my research work in this area and provided me with a lot of literature, equipment and a large number of their advices pending my professional career.

I would also like to express my special gratitude to prof. dr. Hans-Dieter Burkhard, who provided me with a lot of information during my five months stay at the Humboldt University, Berlin in 2002. His great knowledge and excellent experience in the field of case-based reasoning were of extreme value to me.

Finally, I would also like to acknowledge my special thanks to my wife Tatjana for her endless patience and understanding. My warm thanks are also dedicated to my mother Mirjana for her valuable and constant support, encouragement and a great help in solving a number of the English language terms and dilemmas.

Novi Sad, 2009 Vladimir Kurbalija

CONTENTS

Chapter I

INTRODUCTION

Much of the world's supply of data is in the form of time series. During the last few years, there has been an explosion of interest in mining time series data. A number of new algorithms have been introduced to classify, cluster, segment, index, discover rules and detect anomalies/novelties in time series. While many of these different techniques used to solve these problems use a large number of different techniques, they all have one common factor: they require some high level representation of the data rather than the original raw data. These high level representations are necessary as a feature extraction step, or simply to make a storage, transmission and computation of a massive dataset possible. Many of representations have been proposed in literature, including spectral transforms, wavelets transforms, piecewise polynomials, eigenfunctions and symbolic mappings.

A novel time series representation based on splines is proposed in this Thesis. Together with this representation, an original similarity measure specialised for this representation is also proposed. Similarity measure is very important for indexing problem, as well as for other task types. Furthermore, an algorithm for time series forecasting, based on the Case Based Reasoning technology, is also presented in this Thesis. All mentioned concepts are implemented in the system *CuBaGe* (Curve Base Generator).

Speaking in general, the case based reasoning is a problem solving technique, where new problems are solved by adapting solutions that worked for similar problems in the past. The cases, generally, can be represented in a form *(problem, solution)* where the first element represents a description of the problem and the second one a successful solution of the same problem in the past. The basic scenario for mainly all CBR applications looks as follows:

> *In order to find a solution of an actual problem, one looks for a similar problem in an experience base, takes the solution from the past, which is the most similar to the actual problem, and uses it as a starting point to find the solution of the actual problem.*

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 between the parameters of the problem; it just tries to find similar problems (from the past) and to use solutions of them as a solution of an actual problem. So, this approach is extremely suitable for less examined domains – for domains where the rules and connections between the parameters are not known. The second very important advantage is that CBR approach to learning and problem solving is a very similar to human cognitive processes – people take into account and use past experiences to make future decisions.

On the other hand, a curve analysis is a popular area in a current research, which can be illustrated by a large number of papers published recently. This explosion of interest in time series can be explained by a 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 ones are:

- Indexing
- Clustering
- Classification
- Prediction (Forecasting)
- Summarization

- Anomaly Detection
- Segmentation

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.

The 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 a high-level abstraction of the data instead of the original data were developed. Some of the most important are: Discrete Fourier Transform, Discrete Wavelet Transform, Singular Value Decomposition, Piecewise Linear, Piecewise Constant models, Adaptive Piecewise Constant Approximation, Symbolic Aggregate Approximation, etc.

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 an overall shape of the sequence. However, DFT and DWT are only defined for the 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 on the domain and type of the application as well, but it also depends on the chosen representation.

In this Thesis, one combination of curve (time series) representation together with the corresponding similarity/distance measure has been presented. It is decided to lay stress on generality and robustness for account of efficiency. Presented technique is 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 this respective technique from the techniques mentioned above.

Chapter II

FOUNDATIONS OF CBR

Generally speaking, case-based reasoning (CBR) is applied for solving new problems by adapting solutions that worked for similar problems in the past.

This technique is relatively novel and promising. In the past ten years, CBR technique was frequently used in different domains. Furthermore, CBR approach is used in different problem types and systems that require some kind of intelligent behaviour [Burkhard 2001], [Burkhard, Richter 2001], [Bartsch-Spörl 1999].

CBR is the key technique used in this Thesis. Developed system (*CuBaGe*) for time-series analysis is based on this technique. Time-series prediction is realized following all main phases of CBR methodology. In this chapter, some formal or informal definitions of the basic concepts will be given. Definitions are taken from [Lenz et al. 1998].

2.1. KNOWLEDGE

Knowledge can be understood as an informal notion describing something that a human, a formal system or a machine can possibly use in order to perform a certain task (to solve a

problem). In order to use knowledge some entities need to have an access to it and to know how to apply it in solving problems.

The way the knowledge is expressed is the type of knowledge representation, which consists of certain data structures and some additional operators that allow changes on the data structure. The most common data structure in case-based reasoning is the *attribute-value representation*. Every attribute is given by:

- a name A,
- a usually finite set $DOM(A)$ called the domain of the attribute A, and
- a variable x_A, which will hold the value of the attribute.

For a finite set A_i, $1 <= i <= n$, of attributes an *attribute-value vector* is an n-tuple $(a_1, ... ,a_n)$ such that $a_i \in DOM(A_i)$. However, if one wants to deal with incomplete knowledge (which is occurring frequently in case-based reasoning), one must allow that some variables exist without values (value unknown).

2.2. BASIC CONCEPTS

Case-based reasoning is a problem solving technology. The basic scenario for case-based reasoning, from the simplified point of view, looks as follows:

> *In order to find a solution of an actual problem one looks for a similar problem in an experience base, takes the solution from the past and uses it as a starting point to find a solution of the actual problem.*

A general requirement in every knowledge-based system is to make use of the past experience. An experience may be concerned with what was true or false, correct or incorrect, more or less useful. It can be represented by a rule, constraint, some general law or advice or simply by saving a past event. From all of this, the main idea of the case can be obtained. The *case* is some recorded situation where the problem was totally or partially solved [Minor 1998], [Minor 1999], [Minor 2000]. In its simplest form, the case is represented as an ordered pair:

(problem, solution)

The existence of the case means that the corresponding episode happened in the past. This episode contains some decisions that a decision maker finds useful. However, somebody else may not be happy with such a case and neglect it. From this follows that cases must be selected carefully, so different categories of cases can exist: good, typical, important, misleading or unnecessary.

A *case base* is a set of cases, which is usually equipped with some additional structure. A structured case base is usually called a *case memory*.

In many practical applications, one deals with problems with incomplete information. Both, the problem and the solution part in the case, may be incompletely described. In these situations we talk about *incomplete cases*. The case completion is another task that can be solved by using case-based reasoning technology.

The next very important concept in the case-based reasoning is *similarity*. While in classical databases information can be retrieved by using only exact matches, in the case-based reasoning cases can be retrieved by using even inexact matches. The notion of similarity is equivalent to a dual mathematical concept - *distance*.

In the functional way similarity can be defined as a function:

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

where U refers to the universe of all objects, while CB refers to the case base (just those objects which were examined in the past and saved in the case memory). The higher value of the similarity function means that these objects are more similar. The boundary case is $sim(x,x) = 1$, which means that each object is the most similar to itself.

Retrieval is a basic operation in databases and therefore in the case base too. A query to a database retrieves some information by an exact match by using a key, while a query to a case-based reasoning system presents a problem and returns a solution by using inexact matches with the problems from the cases in the case base.

As in databases, trees play a major role in efficient retrieval. Some examples of retrieval structures are: kd-trees (k-dimensional trees), case retrieval nets, discrimination nets, etc.

The simplest way to use retrieved case is simply to take the unchanged solution of that case as the solution to the actual problem. However, in many applications even small differences between the actual and the case problem may require significant modifications to the solution. Making the appropriate changes to the case solution is called a *case adaptation*. A general demand is that the solution of a similar problem should be easily adapted to a solution of an actual problem.

The *knowledge container* is the structural element, which contains some quantity of knowledge. The idea of the knowledge container is totally different from the traditional module concept in programming. While the module is responsible for a certain subtask, the knowledge container does not complete the subtask but contains some knowledge relevant to many tasks. On the other hand, even small tasks require the participation of each container. The concept of the knowledge container is similar to concepts of the nodes and propagation rules in neural networks [Park 2004].

In case-based reasoning we identify the following knowledge containers [Richter 1995]:

- the vocabulary used;
- the similarity measure;
- the case base; and
- the solution transformation.

In principle, each of the mentioned containers can carry almost all knowledge available. From a software engineering point of view there is another advantage of case-based reasoning - the content of the containers can be changed locally. This means that manipulations on one container have little consequences on the other ones. As a consequence, maintenance operations [Iglezakis 2001], [Reinartz 2000], [Wendler 2001] are easier to be performed then on classical knowledge based systems.

The task of Machine Learning is to improve a certain performance by using some experience or instructions. In inductive learning, problems and good solutions are presented to the system. The major desire is to improve a general solution method in every inductive step. Machine Learning methods can be used in order to improve the knowledge containers of a case-based reasoning system (the case base, similarity

measures and the solution transformation). However, one of the greatest advantages of the case-based reasoning system is that it can learn even through the work with users modifying some knowledge containers.

The case based reasoning system has not only to provide solutions to problems but also to take care of some other tasks occurring when it is used in practice. The main phases of the case-based reasoning activities are described in the *CBR-cycle* of [Aamodt, Plaza 1994] in Figure 2.1.

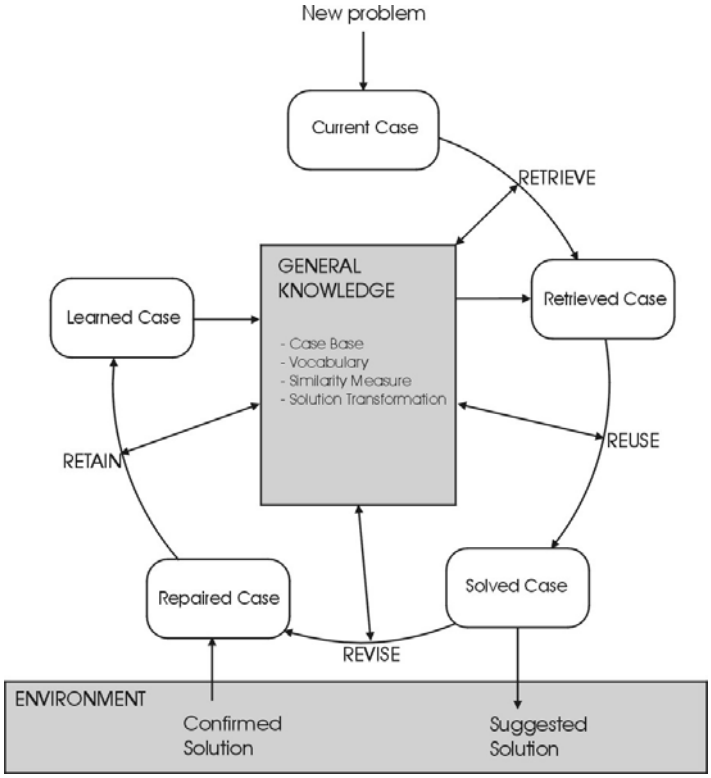

Figure 2.1. *CBR-cycle* of Aamodt and Plaza (1994)

Slika 2.1. *CBR-krug* prema Aamodt-u i Plaza-i (1994)

In the *retrieve* phase the most similar case (or k most similar cases), to the actual problem case, is retrieved, while in the *reuse* phase some modifications to the retrieved case is done in order to provide better solution to the problem (case adaptation). As the case-based reasoning only suggests solutions, there may be a need for a correctness proof or an external validation. That is the task of the phase *revise*. In the *retain* phase the knowledge,

learned from this problem, is integrated in the system by modifying some knowledge containers.

2.3. EXTENDING BASIC CONCEPTS OF CBR

The case-based reasoning technology was developed in the context and in the neighbourhood of problem solving methods, learning methods (Machine Learning, Statistics, Neural Networks) and retrieval methods (Data Bases, Information Retrieval). It has inherited the concepts of "problem" and "solution" and a notion of "similarity" based on the distance.

The basic concepts of the case-based reasoning can be extended in the following way:

- The term *case completion* instead of *solution* will be used (including solutions of problems but also a proposal of intermediate problem solving steps).
- The *cases* will be considered as sets of *information entities* instead of vectors (whereby this approach includes vectors as well as textual documents).
- The term *acceptance* instead of *similarity* will be used, because acceptance includes similarity but also other approaches related to "expected usefulness", "reminds on" etc.

2.3.1. Case Completion

Case-based reasoning is considered as a problem solving method; given a problem we have to find its solution. This leads to a view where cases are split into the problem and the solution part. Given a new problem we search for related problems in the case memory and adapt their solution for the new problem. However, problem solving usually does not start with a complete problem description, which makes the identification of a final solution more difficult.

Nevertheless, the new case is often talked of as a given entity when only the first impression of the underlying task is given. Instead of this, we want to keep attention to the fact that the whole process of completing the task up to the final solution. The consequence is that the resulting case usually depends on a number of decisions. These decisions are initially open; they depend on future human decisions. Depending on different possible decisions, we can end up with different cases.

Most practical tasks are performed as processes with a lot of intermediate steps. Each of these steps could be considered as a single new problem-solving step. The question arises, as to whether we need different sets of cases to support each of these steps. This would mean splitting the whole story of the process into different cases for a later usage by the case-based reasoning. Case completion is an attempt to avoid such approach. A case should be a description of the whole performance of the task with all steps, and it should be useful for later tasks at intermediate situations, too. The main consequence of the case completion is that we do not actually need any formal distinction between the problem and the solution part in the case.

2.3.2. Information Entities

Information entities are atomic constituents of cases and queries. We consider a case as the result of the case completion process. Each step of that process adds some information entities. The current situation during the elaboration of a task is described by the information entities known at the time point. The final case, as it later may appear in the case memory, is a completed set of information entities.

The collected information entities result from the real world (an outcome of the test, a decision in an intermediate design step, etc). They are not a direct result of the case-based reasoning process - case-based reasoning is used to propose the next step (some test, the next design decision etc).

The number of information entities in a case may be variable. It is up to a human decision at which time point the task is finished.

The information entities, which are later used for retrieval, (which appear in the case memory) may be only a subset of the information entities collected during the case completion. These information entities (in the case memory) serve as the indexes for retrieval. The case memory consists of cases, which are sets of such information entities. These cases may then point to related complete descriptions in a collection of "full cases".

The information entity is an atomic part of a case or query. E denotes the set of all information entities in a given domain.

- A case is a set of information entities: $c \subseteq E$.
- The set of cases (in the case memory) is denoted by C, $C \subseteq P(E)$.
- A query is a set of information entities: $q \subseteq E$. This definition of a query differs from standard definition in database terminology. As a consequence of case completition process, the query should have the same structure as the case.

In many applications, the information entities are simply attribute-value pairs. Some examples of information entities are:

<Price, 1000>, <Price, 324>, <Colour, blue>, <Mass, 54 kg>.

We say that the first two information entities are comparable (because they have the same attribute) while the other information entities are not comparable.

This causes a structuring of the set E into disjoint sets E_A, where E_A contains all attribute-value pairs from E for a certain attribute A.

If cases and queries are considered as attribute-value vectors over a finite set of attributes $A_1,\ldots,A_n$, then each case or query may contain at most one information entity from each E_{Ai}.

2.3.3. Acceptance

The general idea is to use the association of information entities for reminding cases with the expectation that these cases are useful for a given query. Usefulness of a case in the case completion process depends on real world circumstances that are not completely known at the retrieval time. This means that usefulness is only a posterior criterion. The retrieval from the case memory will be based on matching of certain information entities. Usefulness of former cases is not restricted to those cases that are similar to a given query for all information entities. Cases may contain information entities that have no counterpart in the query. It is also possible that some information entities of the query are not present in the useful case.

Some special desirable properties of acceptance are following:

P1: A case might be acceptable for a query even if there exist some information entities that are not comparable.

P2: A case might be unacceptable for a query if there exists an unacceptable information entity (a fix budget may forbid expensive offers).

P3: The same information entity may have different importance for different cases (for example information entity <*sex, male*> has different importance in pregnancy testing and in testing for influenza).

P4: The same information entity may have different importance for different queries according to the user's intentions (for example material has different priorities in design queries).

P5: Information entities may not be independent of each other.

In order to provide better understanding of acceptance the *preference relation* ($\geq_q$), over the set of all potential cases, will be defined.

$c' \geq_q c''$ if case c' is preferable to case c'' in regard to the query q.

At the beginning, the definitions of acceptance functions will be given for cases and queries represented as feature vectors (and not as a set of information entities) because it is a more convenient form of representation. Both the queries and the cases are considered as feature vectors $(a_1,\ldots,a_n)$, where a_i specifies the value for the i-th attribute A_i.

Formally, there is no difference between case vectors $c=(c_1,\ldots,c_n)$ and query vectors $q=(q_1,\ldots,q_n)$.

Definition 1. (Global Acceptance Function): *Let $U := dom(A_1) \times \ldots \times dom(A_n)$ denotes the set of all queries and cases. The acceptance of a case for a query is expressed by a global acceptance function*

$$acc : U \times U \rightarrow R \qquad (2.1)$$

such as that a higher value acc(q,c) denotes a higher acceptance of the case c for the query q. The preference relation $\geq_q \subseteq U \times U$ induced by a query $q \in U$ is defined by

$$c' \geq_q c'' \text{ iff } acc(q,c') \geq acc(q,c''). \square \qquad (2.2)$$

Definition 2. (Local Acceptance Functions for Attributes): *A local acceptance function σ_i for the attribute A_i is defined over the domain $dom(A_i)$:*

$$\sigma_i : dom(A_i) \times dom(A_i) \rightarrow R \qquad (2.3)$$

such as that a higher value $\sigma_i(q_i,c_i)$ denotes a higher acceptance of the value c_i (of a case c) for the value q_i (of a query q). □

Global acceptance function can be obtained from local acceptance functions by a related composition function as follows:

Definition 3. (Composite Acceptance Function): *A global acceptance function acc is called composite if it is composed by a composite function $\Phi : R \times \ldots \times R \rightarrow R$ from related local acceptance functions σ_i:*

$$acc((q_1,\ldots,q_n),(c_1,\ldots,c_n)) = \Phi(\sigma_1(q_1,c_1),\ldots,\sigma_n(q_n,c_n)). \square \qquad (2.4)$$

The natural demand is that composition function must be monotonously increasing.

An example for the composition of local acceptance values is given by a weighted sum with only positive weights g_i (because of monotonously increasing composition function):

$$acc((q_1,\ldots,q_n),(c_1,\ldots,c_n)) = \sum g_i \cdot \sigma_i(q_i,c_i) \qquad (2.5)$$

Addition is widely used for the combination of local values. It has an intuitive interpretation concerning acceptance in the sense of "collecting arguments" in favour of something. Positive arguments have positive values, while negative arguments are expressed by negative values. The value 0 does not change the result, so unimportant or unknown attributes can be treated as value 0. Therefore, properties **P1** and **P2** are satisfied. However, a more general combination is necessary to satisfy properties **P3**, **P4** and **P5**.

2.3.4. The general case

In this section the previous concepts will be generalized in order to satisfy the properties **P1**,…,**P5**.

Here, queries and cases are considered as sets of information entities. A *weighted query* is the generalization of this concept.

Definition 4. (Weighted query): *The weighted query assigns an importance value to each information entity by a function:*

$$\alpha_q : E \rightarrow R, \qquad (2.6)$$

where $\alpha_q(e)$ denotes the importance of the information entity e for the query q. □

High values indicate a high importance; negative values indicate the rejection of related cases. The value 0 is used as a neutral element ($\alpha_q(e)=0$, means that information entity e is unimportant to the query q). Of course, values for $\alpha_q(e)$ can be simply taken from the set $\{0,1\}$, where the 0 value means that "e is unimportant for the q", while the 1 value means "e is important for the q".

By using σ (Definition 2.) we can compute the acceptance of the information entity e' from the case for a single information entity e of a query. However, a query may contain several information entities e such that $\sigma(e,e')$ is defined for the single information entity e'. The question is: how these values can be combined to a single value for e' which expresses the resulting acceptance value of e' for that query.

Definition 5. (Local Accumulation Function): *Let $E_e = \{e_1, \ldots, e_n\}$ denote the set of all information entities to which the information entity e is comparable concerning acceptance ($E_e = \{e' \mid \sigma(e',e)$ is defined$\}$). The local accumulation function π_e for e is a function:*

$$\pi_e : \underbrace{R \times \ldots \times R}_{n \_ times} \rightarrow R \qquad (2.7)$$

such that $\pi_e(a_1, \ldots, a_n)$ denotes the accumulated acceptance in e. The values a_i denote the contributions of the information entities $e_i \in E_e$ according to their occurrence in the query q and their local acceptance computed by $\sigma(e_i,e)$.

The contributions are computed by a function:

$$f : R \times R \rightarrow R \qquad (2.8)$$

such that $a_i = f(\alpha_q(e_i),\sigma(e_i,e))$. □

We consider the retrieval of the cases as a process of reminding. Reminding may be of a different strength; cases are in competition for retrieval according to the query. The cases

receiving more reminders of more strength are the winners. The strength (importance, relevance) of reminding for an information entity $e \in c$ is given by a relevance function:

Definition 6. (Relevance Function): *The relevance between information entities and cases is described by a relevance function:*

$$\rho : E \times C \to R. \qquad (2.9)$$

The relevance $\rho(e,c)$ is considered as a measure for the relevance of information entity e for the retrieval of a case c. $\rho(e,c)$ is defined if and only if $e \in c$. □

Negative values $\rho(e,c)$ may be used in the meaning "do not retrieve the case c if one asks for the information entity e".

The acceptance of a case c for the query q is accumulated from the contributions of the information entities $e \in c$ according to their relevancies $\rho(e,c)$. The contributions p_e of the information entities are computed by their accumulation functions π_e as described in the definition 5. The accumulation in the cases is evaluated by Global accumulation function.

Definition 7. (Global Accumulation Function): *The global accumulation function π_c has the form:*

$$\pi_c : \underbrace{R \times \ldots \times R}_{k \_ \text{times}} \to R \qquad (2.10)$$

for $c = \{e_1, \ldots, e_k\}$. The accumulated acceptance of the case c regarding its constituting information entities is then computed by $\pi_c(p_1, \ldots, p_k)$, where p_i is the contribution of the information entity $e_i \in c$. This contribution p_i depends on $\rho(e_i,c)$ and another real value x_i assigned to e_i (x_i is the accumulated local acceptance value computed by $\pi_{ei}(a_1,\ldots,a_n)$ from definition 5.). The contributions p_i are computed by a function:

$$g : R \times R \to R, \qquad (2.11)$$

such that $p_i = g(x_i, \rho(e_i,c))$. □

The global acceptance function, which satisfies properties **P1**,…,**P4**, specified in definition 8, is calculated in the following way:

Definition 8. (Extended Acceptance Function): *Acceptance between weighted queries and cases is expressed by an extended acceptance function:*

$$acc : R^E \times P(E) \rightarrow R. \square \qquad (2.12)$$

The acceptance $acc(\alpha_q,c)$ of a case c for a weighted query α_q can now be accumulated by using the introduced functions:

$$acc(\alpha_q,c) = \pi_c (g (\pi_{e'1} (f (\alpha_q (e_{1,1}), \sigma(e_{1,1},e'_1)),\ldots,f (\alpha_q (e_{1,n1}), \sigma(e_{1,n1},e'_1)), \rho(e_1',c)),$$
$$\ldots$$
$$g (\pi_{e'k} (f (\alpha_q (e_{k,1}), \sigma(e_{k,1},e'_k)),\ldots,f (\alpha_q (e_{k,nk}), \sigma(e_{k,nk},e'_k)), \rho(e_k',c)))$$
$$\text{where } c=\{e'_1,\ldots,e'_k\} \text{ and } E_{e'i} = \{e_{i,1},\ldots e_{i,ni}\} \text{ for } i = 1,\ldots,k. \qquad (2.13)$$

If, for example, we consider f and g as products and π_c and π_e as sums then we get:

$$acc(\alpha_q,c) = \sum_{e'\in c} \rho(e',c) \sum_{e\in E_{e'}} \sigma(e,e') \cdot \alpha_q(e) \qquad (2.14)$$

Here, the properties **P1**,…,**P4** are satisfied, but for satisfaction of the property **P5**, the appropriate selection of the functions f, g, π_c and π_e is needed.

2.4. TYPES OF CBR APPLICATIONS

In this section the basic characteristics for every task type, which can be potentially solved with CBR technology, will be given together with several realized applications [University of Kaiserslautern, CBR homepage]. However, before explaining task types it is necessary to make a difference between *domain* and *task type*.

Domain of application represents the area or discipline in which the CBR technology is used for solving problems. For example, the domains are: mechanical engineering, business administration, medicine, etc. Each domain has its own characteristics, and it strongly influences the choice of data structure for knowledge representation.

Task type represents the kind of problem which has to be solved in some domain. Some of the task types are:

- Classification,
- Diagnosis,
- Configuration,
- Planning,
- Decision Support,
- Information Searching, etc.

Task type determines the type of problems and solutions and, furthermore, the activities for solving problems. There is no one-to-one correspondence between task type and domain. Furthermore, every pair (domain, task type) is possible and it requires its own expertise.

2.4.1. Classification

Classification is a process of determining a class for some elements from a given set, using corresponding functions. The parameters for classification are: universal set U and its subsets $K_i \subseteq U,\ i \in I,$ called classes. A classifier is a function:

$$f: U \to I \qquad (2.15)$$

such that $f(x)=i$ implies $x \in K_i$.

In CBR technology the classifier is defined as a pair $(CB,\ sim)$, where $CB \subseteq U$ is a case base, and sim similarity measure defined on $U \times CB$. If classes of the elements from case base CB are known, the class of the element from universal set $x \in U$ is computed by using sim in the following way:

$$x \in K_i \Leftrightarrow NN(x) \in K_i \qquad (2.16)$$

where $NN(x)$ is the nearest neighbour of x, but in the set CB. Of course, the nearest neighbour can be computed by using k nearest neighbours.

2.4.2. Diagnosis

Diagnosis is considered as a cost sensitive classification with incomplete information [Lenz, Burkhard 1996], [Kamp, Pirk, Burkhard 1996], [Kurbalija 2003]. This means that establishing the diagnosis is only a final step of a diagnostic process. The process of

diagnosis if often interleaved with some additional tests or questions which adds some information entities in order to establish a better diagnosis. This is often called a *differential diagnosis*.

Some commercial applications for diagnosis, realized using CBR technology are:

- **Case Advisor 4 / Webserver** – diagnosis of malfunctions and solving problems for PC computers. This tool uses static database. Case Advisor Webserver is a tool that offers online support to the users. Here is also offered an online support to the users of the cable TV network. Detailed information can be found on: http://www.cs.sfu.ca/~isa/isaresearch.html#systems
- **Case-Based Reasoning in Cardiovascular Disease –** diagnosis of some cardiovascular diseases on the basis of some symptoms. Case base consists of 240 characteristic cases. Also this system supports learning form the experience. Detailed information can be found on: http://medg.lcs.mit.edu/projects/cbr.html
- **MoCas –** diagnosis in technical domains. The main characteristic of this system is that the general knowledge from some technical domain is integrated in the system. Integration of the knowledge enables adaptation and transformation of the cases. Detailed information can be found on: http://www.informatik.uni-kl.de/~lsa/LSABook-English/LSABook-E_20.html
- **SpectroRx Resolution Expert –** tool for diagnosis problems with computer network. Company "ENTERASYS", whose main area is production of network equipment, used CBR technology to make a help desk system which can help clients with various problems. This system saves the expert time because experts help clients only when system can not, and the rest of the time spends on more complex tasks. Detailed information can be found on: http://www.cabletron.com/products/items/SA-CSI1016/
- **Cassiopee** – a software system to support fault diagnosis of the CFM 56-3 aircraft engine for the Boeing 737. The system uses data-mining techniques known as induction and case based reasoning which exploit failure descriptions that are stored in a case base. A first prototype using the induction technique applied on an initial case base has been achieved [Heider 1996].
- **ICARUS** – Diagnose of locomotives (Intelligent Case-based Analysis for Railroad Uptime Support). ICARUS is a case-based reasoning system for diagnosing locomotive faults using fault messages as input [Varma 1999].
- **CASEY** – an expert system in the domain of heart failure diagnosis that combines case-based and rule-based reasoning techniques. The system uses three steps: a search for similar cases, a determination of differences and their evidences, and a transfer from the diagnoses of similar cases or - if the differences are too important - an attempt to explain and repair them. If no similar case can be found or if all repair attempts fail, CASEY uses the rule-based domain theory [Koton 1988].

- **PSIQ** – gives diagnostic and therapeutic advice in the domain of mental disorders. After a language treatment, component has delivered ICD-10 categories from the patient's history and grouped them, concerning different aspects. These categories can be used to search for similar cases which are shown to the user (including their diagnoses and treatments) [Schwartz 1997].
- **GS.52** –a diagnostic support system for dysmorphic syndromes. Such a syndrome involves a non-random combination of different disorders. The major problems are a high variability of the syndromes (hundreds), a high number of case features (between 40 and 130) and continuous modifications of the knowledge about dysmorphic syndromes [Gierl, Stengel-Rutkowski 1992]; [Gierl, Stengel-Rutkowski 1994].
- **MERSY** – a system for rural health care workers. This system utilizes the most convincing advantage of CBR - the relocation of expertise. This is especially valuable in countries where the health care workers are rare and perform with only modest skills. The knowledge-base can be automatically adapted to the special health care problems of a region by simply using the CBR system [Opiyo 1995].

2.4.3. Configuration and Design

Configuration can be understood as the construction of the artifact from the given set of components, such that certain conditions are satisfied. *Design* introduces some degree of creativity because some components of the artifact are not known during design process. Depending on the degree of creativity required 3 types of design can be distinguished: routine design, innovative design and creative design. In real world applications the solution gained from CBR configuration or design system can almost never remain unmodified – the adaptation phase is necessary.

Some commercial applications for configuration and design, realized by using CBR technology are:

- **AIDA** (Artificial Intelligence supported Design of Aircraft) **–** tool for helping in airplane design. This tool helps in the first phase of design process – conceptual design. Human designer can pay attention on more creative tasks while the system keeps attention on less creative tasks. For realization of this tool several artificial intelligence approaches are used: Constraint-Based Reasoning, Case-Based Reasoning and Rule-Based Reasoning. The intention of the authors is to use the concept of AIDA in other domains such as: car or ship design. Detailed information can be found on: http://www.kbs.twi.tudelft.nl/Research/Projects/AIDA/
- **Archie –** tool for helping in conceptual building design. The aim of this tool is to make use of past experience in design of public buildings in order to avoid some

previous mistakes in design process. Detailed information can be found on: http://www.cc.gatech.edu/aimosaic/faculty/kolodner/archie.html

- **BRUSH –** tool for designing bathrooms for the invalids. Cases in this tool represent episodes in design bathrooms for invalids. System, on the basis of those cases, designs a new bathroom which satisfies certain properties. Detailed information can be found on: http://www.arch.su.edu.au/~kate/BathRedesign/Guestbookii/Welcome.html
- **CADET –** tool for conceptual design of electro-mechanical parts. CADET consists of several subsystems. Case-based reasoning and model based reasoning are integrated in this system. Detailed information can be found on: http://www.cs.cmu.edu /afs/cs.cmu.edu/project/cadet/ftp/docs/CADET.html
- **CBRTeam –** multiagent system for design of steam condenser. This system consists of 3 agents: `motor-agent`, `pump-agent` and `vbelt-agent` whish are responsible for design of motor, pump and belt respectively. When a user makes a specification these agents, in cooperation and on the basis of available parts, tries to construct suitable condenser. Detailed information can be found on: http://ksi.cpsc.ucalgary.ca/KAW/KAW96/prasad/
- **Clavier** – a case-based reasoning (CBR) system that assists in determining efficient loads of composite material parts to be cured in an autoclave. Clavier's central purpose is to find the most appropriate groupings and configurations of parts (or loads) in order to maximize autoclave throughput while assuring that parts are properly cured. Clavier uses case-based reasoning to match a list of parts that need to be cured against a library of previously successful loads and suggest the most appropriate next load [Hinkle and Toomey 1994].

2.4.4. Planning

Planning covers a great variety of tasks. Here, planning will be restricted on the *action planning*. The problem is to find a sequence of actions which transforms a given initial situation into a desired goal situation. The number of selected actions is not restricted but the set of available actions is fixed. For CBR planning, the reuse aspect is important because the retrieved plans have to be adapted.

Some commercial applications for planning, realized by using CBR technology are:

- **Bioplan** – bioprocess planning in medicine production. Detailed information can be found on: http://www.vtt.fi/bel/bio/process/bioplan.htm
- **Knowledge Based Mashing –** process planning in beer production. Detailed information can be found on: http://www.vtt.fi/bel/bio/process/mashplan.htm
- **Prodigy –** domain independent action planning. Prodigy is architecture for learning and planning. At this moment Prodigy supports different concepts: example based learning, partial calculations, graphical learning, automatic abstraction, initial planning, CBR in many different domains. Detailed

information can be found on: http://www.cs.cmu.edu/afs/cs.cmu.edu/project/prodigy/Web/prodigy-home.html

- **CHARADE –** planning of human and equipment resources in state of emergency (fire, flood….). Detailed information can be found on: http://sra.itc.it/projects/charade/
- **FLORENCE** – a health care planning for nursing. The current patient is compared to a similar previous patient for whom the progression of the health status is known. Similar patients are searched for first concerning the overall status and subsequently concerning the individual health indicators. [Bradburn and Zeleznikow 1993].
- **ICONS** – an antibiotics therapy adviser for intensive care patients who develop an infection as additional complication. The identification of the pathogen that causes the infection needs at least 24 hours in the laboratory. In contrast to normal patients, where physicians usually can wait for the results from the laboratory, intensive care patients need an immediate introduction of an appropriate antibiotics therapy. As the real pathogen is still unknown, a spectrum of probable pathogens has to be calculated. The aim of ICONS is to give rapid antibiotic therapy advice. CBR techniques are used to speed up the process of finding suitable antibiotics therapies and to update those parts of the knowledge-base that are modified frequently [Gierl et al. 2003].
- **CAMP** – prototype for daily menu planning which meets individual, nutritional and personal preference requirements [Kovacic et al. 1992].
- **Orca** – an intelligent mission controller for autonomous underwater vehicles (AUVs). Orca uses procedural schemas, which are like hierarchical plans, to control its actions. It uses contextual schemas, which are similar to generalized cases, to ensure that its behaviour is tailored to its problem-solving situation [Turner 1995].
- **CaPER** – Case based planer is based on a massively frame-based AI language (PARKA) and can do extremely fast retrieval of complex cases from a large, unindexed memory. The ability to do fast, frequent retrievals has many advantages: indexing is unnecessary; very large casebases can be used; and memory can be probed in numerous alternate ways, allowing more specific retrieval of stored plans that better fit a target problem with less adaptation. The flexible nature of case retrieval is also being exploited by treating the retrieval task itself as a planning problem, distinct from the overall planning task in which it is embedded. This "planning to retrieve" approach was motivated by the results of psychologists' studies of human long-term memory [Kettler et al. 1994].

2.4.5. Decision support

The task of a system for decision support system is to help to the person who has to make some important decisions, and not to totally replace him [Ivanović et al. 2002],

[Kurbalija, Ivanović 2003]. The answer of the system is usually not the solution of the problem but some advice or some useful piece of information.

Some commercial applications for decision support, realized by using CBR technology are:

- **Aircraft Conflict Resolution –** application which helps in flight control. This tool helps in solving conflicts in airplane traffic. The great significance of this application is that authors devoted a big effort in representing different kinds of cases. Detailed information can be found on: https://www.cs.tcd.ie/research_groups/aig/old_pages/
- **ALSTOM –** tool for better organization of trains in order to reduce costs. Detailed information can be found on: http://www.acknosoft.com/alstom.html
- **ANSALDO –** maintenance of subway in Napul. Detailed information can be found on: http://www.acknosoft.com/ansaldo.html
- **CBR Job Agent –** job search on the basis of entered skills. Detailed information can be found on: http://minsk.informatik.uni-kl.de:8100/launch/JobbQueryInterface
- **DESSERT –** decision support in obliging management. Detailed information can be found on: http://www.broadcom.ie/partners/acts/race/dessert/dessert.html
- **FormTool** - Plastics Color Matching. Since 1994 GE Plastics has employed a case-based reasoning tool that determines colour formulas which match requested colours. This tool has saved GE millions of dollars in productivity and material (i.e. colorant) costs [Cheetham 2005].
- **Vidur** –a CBR based Advisory System for farmers of North-East India. Vidur is initially developed for weed control and paddy variety selection with the help of agricultural experts. Detailed information can be found on: http://www.cdacmumbai.in/

2.4.6. Information Searching

Inexact matches (finding similar documents when the exact document doesn't exist) are vital here, so the CBR is a natural technique in this area [Lenz, Hübner 1998], [Lenz 1998], [Minor, Carlos 2000]. Since exponential growth of information this is the promising area for CBR, because CBR has ability to learn constantly and also has a good mechanism for indexing huge databases.

Some commercial applications for information searching, realized by using CBR technology are:

- **Broadway –** intelligent web browser. The system tries to use knowledge learned from some previous searches of one group of users. This browser tracks the user in his browsing, tries to discover his goals and proposes some potentially interesting documents. Detailed information can be found on: http://www-sop.inria.fr/aid/broadway/
- **CaBaTa –** virtual tourist agency. On the basis of user's demands, system suggests the most similar offers. Also, the system contains the component for selling airplane tickets. Detailed information can be found on: http://www.reiseboerse.com/
- **Entree –** tool for finding optimal restaurant in Chicago. Detailed information can be found on: http://infolab.cs.uchicago.edu/entree/
- **RECALL –** automatic saving and retrieving 'learned lesions' in "NASA Goddard Space Flight Center". Detailed information can be found on: http://aaaprod.gsfc.nasa.gov/TEAS/DavidAha/DavidAhaPres/index.htm
- **FAQ Finder –** retrieving 'frequently asked questions'. Detailed information can be found on: http://infolab.cs.uchicago.edu/faqfinder/
- **SMART** – Support management automated reasoning technology for Compaq customer service. Compaq's call-logging system which include a problem-resolution component that assists customer support personnel in determining the resolution to a customer's questions and problems [Acorn and Walden 1992].
- **STC** - Support the Customer. This is a Case-based reasoning system for General Electric appliance customer support. The system was created to support customers who purchased appliances from General Electric. When a customer calls General Electric for help, a call-taker uses the system to diagnose the problem and step the customer through its solution. The system has been in use by 300 call-takers since 1999. It has resulted in a 20 percent increase in the probability the customer's problem can be solved over the phone [Cheetham and Goebel 2007].
- **ProtoISIS** – provide an intelligent retrieval for image studies in medical domain. The features of a case are clinical indications and questions which have to be answered. A semantic network relates cases, features and imaging procedures to built explanations for the user's questions. Relations in the network are one of the following: "feature of", "causes", "has exemplar", or "visualizes". ProtoISIS does not use a similarity measure or adaptation of former cases [Kahn and Anderson 1994].

2.4.7. General Frameworks and Tools

CBR has a relatively long history and many practical real world applications have been developed using this methodology. On the other hand, much of the research effort was directed in the development of general frameworks and tools. These frameworks are designed to be domain and task type independent. The purpose of these frameworks is

to maximally simplify the process of creating a real world CBR application. Some general frameworks and tools are listed below:

- **Creek/TrollCreek** – a knowledge-intensive case-based reasoning framework and tool. By combining general, abstracted knowledge and concrete problem episodes in the form of cases, the Creek approach aims to act as an intelligent assistant in a wide variety of tasks. TrollCreek is a knowledge model editor for the Java-version of the Case-Based Reasoning system CREEK. Detailed information can be found on: http://creek.idi.ntnu.no/
- **jCOLIBRI** – an object-oriented framework for building CBR. The underlying architecture of jCOLIBRI consists of two layers: The design layer and the developer layer. The design layer provides tools guiding users through the configuration process and explaining the components' behaviour within the application. The developer layer provides basic Java components, which are required to create a CBR application from a developer's point of view. Detailed information can be found on: http://gaia.fdi.ucm.es/projects/jcolibri/
- **myCBR** – an open source CBR tool, which focuses on domain and similarity modelling for case retrieval. Its intended use is in research and education, as well as for rapid prototyping of applications. Detailed information can be found on: http://www.mycbr-project.net/
- **IUCBRF** – an open source framework for CBR system development, which is implemented in Java and developed at the Indiana University. The framework is designed to facilitate fast and modular development of CBR systems as well as providing a foundation for code sharing by those who are developing CBR systems. Detailed information can be found on: http://www.cs.indiana.edu/~sbogaert/CBR/
- **e:IAS** – empolis Information Access Suite. e:IAS uses CBR as its underlying methodology and provides a client-server based knowledge provision component, the Knowledge Server, and a knowledge modelling component, the Creator, for developing knowledge management systems for a variety of commercial application domains. Detailed information can be found on: http://www.empolis.com/home.html

Chapter III

TIME-SERIES ANALYSIS

In statistics, signal processing, and many other fields, a time series is a sequence of data points, measured typically at successive times, spaced at (often uniform) time intervals. Time series analysis comprises methods that attempt to understand such time series, often either to understand the underlying context of the data points (where did they come from? what generated them?), or to make forecasts (predictions) [Wikipedia].

A time-series database consists of sequences of values or events obtained over repeated measurements of time. The values are typically measured at equal time intervals (e.g., hourly, daily, weekly). Time-series databases are popular in many applications, such as stock market analysis, economic and sales forecasting, budgetary analysis, utility studies, inventory studies, yield projections, workload projections, process and quality control, observation of natural phenomena (such as atmosphere, temperature, wind, earthquake), scientific and engineering experiments, medical treatments etc.

Time series data contain a great part of the world's supply of data which can be illustrated by following quotation [Tufte 1983]:

> *A random sample of 4,000 graphics from 15 of the world's newspapers published from 1974 to 1980 found that more than 75% of all graphics were time series.*

Given the ubiquity of time series data, and the exponentially growing sizes of databases, there has recently been an explosion of interest in time series Data Mining.

During the research, indicated by different practical needs, many of the task types were separated. Some of the most common are [Ratanamahatana et al. 2005]:

- **Classification**: Given an unlabeled time series *Q*, assign it to one of two or more predefined classes [Geurts 2001]; [Keogh and Pazzani 1998].
- **Indexing** (Query by Content): Given a query time series *Q*, and some similarity/dissimilarity measure *D(Q, C)*, find the most similar time series in database DB [Chakrabarti et al. 2002]; [Faloutsos et al. 1994]; [Kahveci and Singh 2001]; [Popivanov et al. 2002].
- **Clustering**: Find natural groupings of the time series in database DB under some similarity/dissimilarity measure *D(Q, C)* [Aach and Church 2001]; [Debregeas and Hebrail 1998]; [Kalpakis et al. 2001]; [Keogh and Pazzani 1998].
- **Prediction** (Forecasting): Given a time series *Q* containing *n* data points, predict the value at time *n + 1*.
- **Summarization**: Given a time series *Q* containing *n* data points where *n* is an extremely large number, create a (possibly graphic) approximation of *Q* which retains its essential features but fits on a single page, computer screen, etc. [Indyk et al. 2000]; [Wijk and Selow 1999].
- **Anomaly Detection** (Interestingness Detection): Given a time series *Q*, assumed to be normal, and an time series *R*, find all sections of *R* which contain anomalies or "surprising/interesting/unexpected" occurrences [Guralnik and Srivastava 1999]; [Keogh et al. 2002]; [Shahabi et al. 2000].
- **Segmentation**: (a) Given a time series *Q* containing *n* data points, construct a model *Q'*, from *K* piecewise segments ($K << n$), such that *Q'* closely approximates *Q* [Keogh and Pazzani 1998]; (b) Given a time series *Q*, partition it into *K* internally homogenous sections [Guralnik and Srivastava 1999].

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.

Time series analysis is the main application area of the system described in this thesis. For that reason, this chapter describes the most important concepts and techniques used in time series analysis. In the first section the time-series similarity will be described in detail. More detailed description of task types in time-series analysis will be given in

section 3.2, while the discussion about time-series representation and dimensionality reduction techniques will be given in section 3.3.

3.1. TIME SERIES SIMILARITY MEASURES

Unlike normal database queries, which find data that match the given query exactly, a similarity search finds data sequences that differ only slightly from the given query sequence. Given a set of time-series sequences, *S*, there are two types of similarity searches: subsequence matching and whole sequence matching. Subsequence matching finds the sequences in *S* that contain subsequences that are similar to a given query sequence *x*, while the whole sequence matching finds a set of sequences in *S* that are similar to each other (as a whole). Subsequence matching is a more frequently encountered problem in applications.

Frequently, it is more convenient to compute distance between time series, and then from distance to compute similarity. For example, a simple function to compute similarity from distance could be:

$$sim\ (x, y) = \frac{1}{1 + dist\ (x, y)} \qquad (3.1)$$

Sometimes, it is not necessary to compute similarity at all. For example, in nearest neighbour search, *k* nearest neighbours can be selected by using minimum distance (instead of maximum similarity). Therefore, the similarity and distance will be treated equally in the rest of this thesis.

The distance measure between two objects $D(O_1,O_2)$ must satisfy the following properties:

- **Symmetry** – $D(A,B) = D(B,A)$
- **Constancy of Self-Similarity** – $D(A,A) = 0$
- **Positivity** – $D(A,B) = 0$ *If* $A = B$
- **Triangular Inequality** – $D(A,B) \leq D(A,C) + D(B,C)$

Frequently, it is necessary to make some simple transformations on time series before computing similarity/distance. The main reason is that the distance between two "raw" time series could be very large although their overall shape is very similar. This is the consequence of some distortions in time series. The task of the pre-processing is to remove these distortions. Some of the most common pre-processing tasks are:

- **Offset Translation.** This transformation is used when time series have different offsets, i.e. when time series are located on different values on *x*-axis. Two time series could be very similar but the distance between them can be very big, which is the consequence of large offset. A very intuitive offset translation transformation is the subtraction the mean value from each time series.
- **Amplitude Scaling.** This transformation is applied when several time series have similar shape but with different amplitudes. The objective of this transformation is to normalize the amplitude of all time series. Most commonly used transformation for amplitude scaling is: subtract the mean value from time series and then divide it with its standard deviation.
- **Removing Linear Trend.** Sometimes several time series have the same behaviour but with different trends. If the behaviour of time series is the most important aspect in desired application the trend could be removed using this transformation. The intuition behind removing linear trend looks like this: Fit the best fitting straight line to the time series, and then subtract that line from the time series.
- **Removing Noise.** Very often time series contains some amount of noise. This noise can significantly change the value of similarity/distance measure. The noise can be removed using different smoothing methods. Smoothing methods try to average every datapoint in the time series with its neighbours. Some smoothing methods like Moving average and Exponential smoothing are described in Chapter 4.

These pre-processing tasks can greatly improve the performance of time series applications by removing different kinds of distortions. However, sometimes the distortions are the most interesting thing about the data and in that case some of these transformations should not be used.

3.1.1. The Minkowski Metrics

The Minkovski metrics is the oldest and the most common distance measure in time series analysis. For two sequences of the same length $Q = (q_1, q_2, \ldots, q_n)$ and $C = (c_1, c_2, \ldots, c_n)$ the Minkowski metrics is given by:

$$D(Q,C) = \sqrt[p]{\sum_{i=1}^{n}(q_i - c_i)^p} \qquad (3.2)$$

This is also considered as L_p norm.

More intuitive aspect is: Assume that both time sequences are of the same length n, we can view each sequence as a point in n-dimensional Euclidean space, and define the dissimilarity/distance between sequences C and Q as $D(C, Q) = Lp(C, Q)$, i.e. the distance between the two points measured by the L_p. For different values of p, different distance measures can be obtained:

- $p = 1$ **Manhattan** (Rectilinear, City Block)
- $p = 2$ **Euclidean**
- $p = \infty$ **Max** (Supremum, "sup")

Figure 3.1 shows the intuition behind the Minkowski metrics. The distance between two time series is computed as a sum of distances between corresponding points of the sequences.

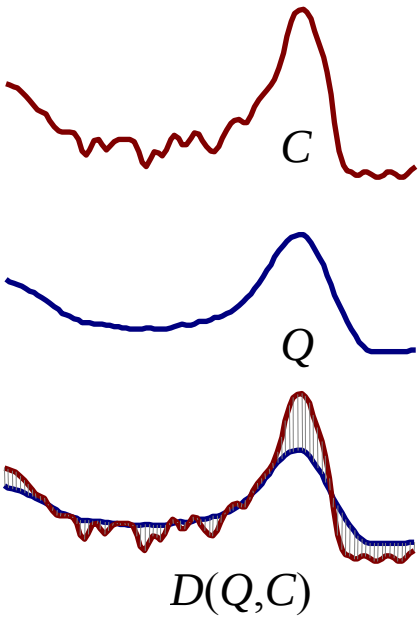

Figure 3.1. Computation of the Minkowksi metrics

Slika 3.1. Izračunavanje metrike Minkovskog

Frequently, instead of Euclidean distance its optimized variation is used – *Squared Euclidean distance*:

$$D_{squared}(Q,C) = \sum_{i=1}^{n}(q_i - c_i)^2 \qquad (3.3)$$

This optimization helps with CPU time, because the squared root is not calculated. Euclidean distance and Squared Euclidean distance are equivalent in the sense that they return the same rankings, clusterings and classifications.

One of the most common modifications of Euclidian distance is *Weighted Euclidian distance*. For two time series $Q = (q_1, q_2, \ldots, q_n)$ and $C = (c_1, c_2, \ldots, c_n)$ and a vector of weights $W = (w_1, w_2, \ldots, w_n)$ the Weighted Euclidian distance is given by:

$$D(Q,C,W) = \sqrt{\sum_{i=1}^{n} w_i(q_i - c_i)^2} \qquad (3.4)$$

The intuition behind this is that for some queries different parts of the sequence are more important. Weights in weight vector W can be specified by an expert from a given domain or can be automatically computed from a training data. The most common algorithm for computing weights is *Relevance Feedback* [Wu et al. 2000], which also requires a participation of the user or expert. The main idea in Relevance Feedback is the reformulation of a search query in response to feedback provided by the user for the results of previous versions of the query.

The Minkowski metrics (especially Euclidean distance) is simple to understand and easy to compute, which has ensured that the Euclidean distance is the most widely used distance measure for similarity search [Agrawal et al. 1993]; [Chan and Fu 1999]; [Faloutsos et al. 1994]. However, one major disadvantage is that it is very brittle; it does not allow for a situation where two sequences are alike, but one has been "stretched" or "compressed" in the Y-axis. For example, a time series may fluctuate with small amplitude between 10 and 20, while another may fluctuate in a similar manner with larger amplitude between 20 and 40. The Euclidean distance between the two time series will be large. This problem can be solved easily with pre-processing tasks: offset translation and amplitude scaling, which requires normalizing the sequences before applying the distance operator.

However, even after normalization, the Euclidean distance measure may still be unsuitable for some time series domains since it does not allow for acceleration and

deceleration along the time axis. This problem can generally be handled by *Dynamic Time Warping* distance measure.

3.1.2. Dynamic Time Warping

In some time series domains, a very simple distance measure such as the Euclidean distance will suffice. However, it is often the case that the two sequences have approximately the same overall component shapes, but these shapes do not line up in X-axis. Figure 3.2 shows this with a simple example. In order to find the similarity between such sequences or as a pre-processing step before averaging them, we must "warp" the time axis of one (or both) sequences to achieve a better alignment. Dynamic Time Warping (DTW) is a technique for effectively achieving this warping.

In [Berndt and Clifford 1996], the authors introduce the technique of dynamic time warping to the Data Mining community. Dynamic time warping is an extensively used technique in speech recognition and many other domains, and allows acceleration/deceleration of signals along the time dimension. The basic idea behind Dynamic Time Warping will be described here.

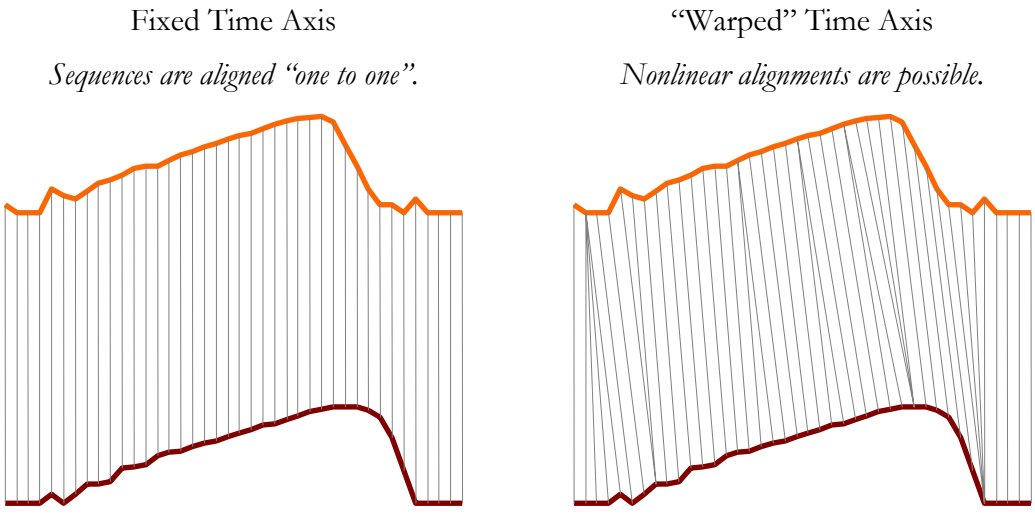

Figure 3.2. Fixed and warped measures

Slika 3.2. Fiksne i „iskrivljene“ mere

For two sequences (of possibly different lengths), $Q= (q_1, q_2, \ldots, q_n)$ and $C= (c_1, c_2, \ldots, c_p)$ a straightforward algorithm for computing the Dynamic Time Warping distance uses a bottom-up dynamic programming approach, where the smaller sub-problems $D(i, j)$ are first determined, and then used to solve the larger sub-problems, until $D(p,n)$ is finally achieved, as illustrated in Figure 3.3.

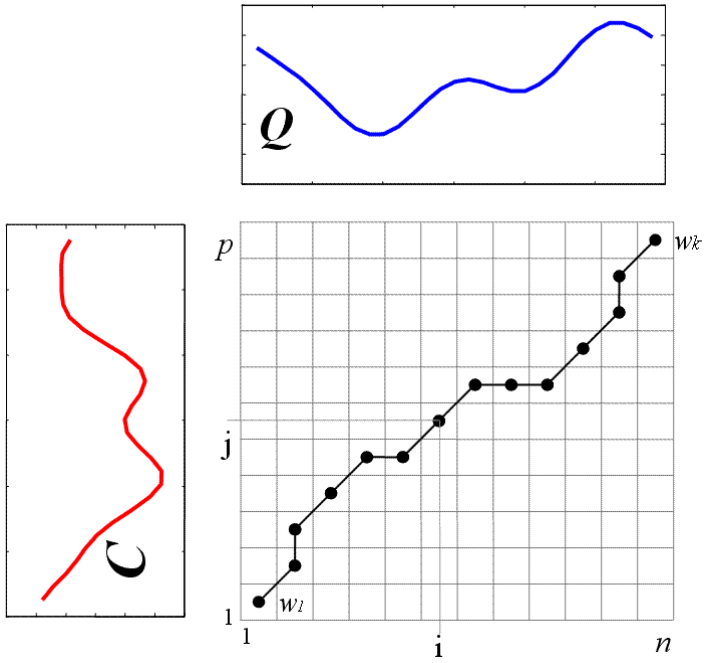

Figure 3.3. Using warping matrix for computing distance

Slika 3.3. Korišćenje matrice „iskrivljenosti“ za izračunavanje rastojanja

To align/match the sequences, we construct a warping matrix, and search for the optimal warping path. Every possible mapping from Q to C can be represented as a warping path in the search matrix. We simply want to find the cheapest one and that is the task of dynamic programming. Although there are exponentially many such paths, we can find one in only quadratic time using dynamic programming.

Although this dynamic programming technique is impressive in its ability to discover the optimal of an exponential number alignments, a basic implementation runs in $O(pn)$ time. In [Ratanamahatana and Keogh 2004], the authors introduce a novel framework based on a learned warping window constraint to further improve the classification accuracy.

The speed up of the DTW calculation by utilizing the lower bounding technique was introduced in [Keogh 2002].

3.1.3. Longest Common Subsequence Similarity

The longest common subsequence similarity measure, or LCSS, is a variation of edit distance used in speech recognition and a text pattern matching. The basic idea is to match two sequences by allowing some elements to be unmatched. The advantage of the LCSS method is that some elements may be unmatched or left out (e.g. outliers), while in the Euclidean distance and DlW, all elements from both sequences must be used, even the outliers.

For example, for two sequences $C = (3, 2, 5, 7, 4, 8, 10, 7)$ and $Q = (2, 5, 4, 7, 3, 10, 8, 6)$ the longest common subsequence is $LCS = (2, 5, 7, 10)$ because this is the longest sequence of elements which appear in both time series in particular order.

More formally, let $C = (c_1, c_2, \ldots, c_m)$ and $Q = (q_1, q_2, \ldots, q_n)$ be two sequences of length m and n, respectively. Let $L(i, j)$ denote the length of longest common subsequences $(c_1, \ldots, c_i)$ and $(q_1, \ldots, q_j)$. $L(i, j)$ may be recursively defined as follows:

IF $c_i = q_j$ *THEN*

$L(i, j) = 1 + L(i - 1, j - 1)$

ELSE

$L(i, j) = max\{L(i - 1, j), L(i, j - 1)\}$

The dissimilarity between C and Q is then defined as:

$$LCSS(C,Q) = \frac{m+n-2l}{m+n} \qquad (3.5)$$

where l is the length of the longest common subsequence. Intuitively, this quantity determines the minimum (normalized) number of elements that should be removed from and inserted into C to transform C to Q. As with dynamic time warping, the LCSS measure can be computed by dynamic programming in $O(mn)$ time.

With time series data, the requirement that the corresponding elements in the common subsequence should match exactly is rather rigid. This problem can be solved by allowing

some tolerance $\varepsilon > 0$ when comparing elements. Thus, two elements c and q are said to match if $c(1- \varepsilon) < q < c(1 + \varepsilon)$.

Two most important modifications of LCSS incorporate local and global scaling of time series:

- **LCSS with Local Scaling** [Agrawal et al. 1995]: The basic idea is that two sequences are similar if they have enough nonoverlapping time-ordered pairs of contiguous subsequences that are similar. Two contiguous subsequences are similar if one can be scaled and translated appropriately to approximately resemble the other. The scaling and translation function is local, i.e. it may be different for other pairs of subsequences.
- **LCSS with Global Scaling** [Bollobas et al. 2001]: Instead of different local scaling functions that apply to different portions of the sequences, a simpler approach is to try and incorporate a single global scaling function with the LCSS similarity measure. The basic idea is that two sequences C and Q are similar if there exists constants a and b, and long common subsequences C' and Q' such that Q' is approximately equal to $aC' + b$. The scale+translation linear function (i.e. the constants a and b) is derived from the subsequences, and not from the original sequences. Thus, outliers cannot "spoil" the scale+translation function.

3.1.4. Probabilistic methods

A different approach to time-series similarity is the use of a probabilistic similarity measure. While previous methods were "distance" based, some of these methods are "model" based. Since time series similarity is naturally a fuzzy problem, the probabilistic methods are well suited for handling noise and uncertainty. They are also suitable for handling scaling and offset translations. Finally, they provide the ability to incorporate prior knowledge into the similarity measure. However, it is not clear whether other problems such as time-series indexing, retrieval and clustering can be efficiently accomplished under probabilistic similarity measures.

The approach described in [Ge and Smyth 2000] looks as follows: Given a sequence C, the basic idea is to construct a probabilistic generative model M_C, i.e. a probability distribution on waveforms. Once a model M_C has been constructed for a sequence C, we can compute similarity as follows: Given a new sequence pattern Q, similarity is measured by computing $p(Q|M_C)$, i.e. the probability that M_C generates Q.

3.1.5. General Transformations

Recognizing the importance of the notion of "shape" in similarity computations, an alternate approach was undertaken by [Jagadish et al. 1995]. In this paper, the authors describe a general similarity framework involving a transformation rules language. Each rule in the transformation language takes an input sequence and produces an output sequence, at a cost that is associated with the rule. The similarity of sequence C to sequence Q is the minimum cost of transforming C to Q by applying a sequence of such rules. The actual rules language is application specific.

3.2. TASK TYPES IN TIME SERIES ANALYSIS

Most classic Machine Learning and Data Mining algorithms do not work well on time series data due to their unique structure and the large volume of data. It is often the case that each individual time series has a very high dimensionality, high feature correlation, and a large amount of noise [Chakrabarti et al. 2002], which presents a difficult challenge in time series Data Mining tasks. Whereas classic algorithms assume relatively low dimensionality (for example, a few measurements such as "height, weight, blood sugar, etc."), time series Data Mining algorithms must be able to deal with dimensionalities in the hundreds or thousands. The problems created by high dimensional data are more than simple computation time considerations: the meanings of normally intuitive terms such as "similar to" and "cluster forming" become unclear in a high dimensional space. The reason is that as the dimensionality increases, all objects become essentially equidistant to each other, and thus classification and clustering lose their meaning. This surprising result is known as the "curse of dimensionality" [Aggarwal et al. 2001]. For this reason, almost all time series Data Mining algorithms avoid operating on the original "raw" data. Instead, they consider some higher-level representation or abstraction of the data. As a consequence, tasks in time series analysis require unique expertise and techniques comparing to traditional Data Mining tasks.

3.2.1. Classification

Classification is perhaps the most familiar and most popular Data Mining technique. Examples of the classification applications include image and pattern recognition, spam

filtering, medical diagnosis, and detecting malfunctions in industry applications. Classification maps input data into predefined groups. It is often referred to as a *supervised learning*, as the classes are determined prior to examining the data (a set of predefined data is used in a training process and learn to recognize patterns of interest). Pattern recognition is a type of classification where an input pattern is classified into one of several classes based on its similarity to these predefined classes.

Two most popular methods in time series classification include the Nearest Neighbour classifier and Decision trees. The Nearest Neighbour method applies the similarity measures of the object to be classified, to determine its best classification based on the existing data that has already been classified. For the decision tree, a set of rules are inferred from the training data, and this set of rules is then applied to any new data to be classified. However, even though decision trees are defined for real data, attempting to apply raw time series data could be a mistake due to its high dimensionality and noise level that would result in deep, bushy tree. Instead, some researchers suggest representing time series as a Regression Tree to be used in the Decision Tree training [Geurts 2001].

3.2.2. Indexing (Query by Content)

Indexing includes a sequence matching task which is usually divided into two categories [Faloutsos et al. 1994]; [Keogh et al. 2001]:

- **Whole Matching**: a query time series is matched against a database of individual time series to identify the ones similar to the query.
- **Subsequence Matching**: a short query subsequence time series is matched against longer time series by sliding it along the longer sequence, looking for the best matching location.

Given a database of sequences, the simplest way to find the closest match to a given query sequence Q, is to perform a linear or sequential scan of the database. Each sequence is retrieved from disk and its distance to the query Q is calculated according to the pre-selected distance measure. After the query sequence is compared to all of the sequences in the database, the one (or k) with the smallest distance is returned to the user as the closest match.

This brute-force technique is costly to implement, first – because it requires many accesses to the disk and second – because it operates on the raw sequences, which can be quite long. Therefore, the performance of a linear scan on the raw data is typically very costly.

A more efficient implementation of the linear scan would be to store two levels of approximation of the data: the raw data and their compressed version. Now the linear scan is performed on the compressed sequences and a *lower bound* to the original distance is calculated for all the sequences. The raw data are retrieved in the order suggested by the lower bound approximation of their distance to the query. After that, the similarity of the raw data can be computed only for the candidates retrieved by a lower bound approximation.

A more efficient way to perform similarity search is to utilize an index structure that will cluster similar sequences into the same group, hence providing faster access to the most promising sequences. By using various pruning techniques, indexing structures can avoid examining large parts of the dataset, while still guaranteeing that the results will be identical with the outcome of the linear scan. Indexing structures can be divided into two major categories: **vector based** (R-tree, kd-B-tree, quad-tree) and **metric based** (VP-tree, M-tree).

3.2.3. Clustering

Clustering is similar to classification that categorizes data into groups. However, these groups are not predefined, but rather defined by the data itself, based on the similarity between time series. It is often referred to as an *unsupervised learning*, since the groups/classes are not known in advance. The clustering is usually accomplished by determining the similarity among the data on predefined attributes. The most similar data are grouped into clusters, but the clusters themselves should be very dissimilar. And since the clusters are not predefined, a domain expert is often required to interpret the meaning of the created clusters. The two general methods of time series clustering are:

- **Partitional Clustering:** typically uses the K-means algorithm (or some variant) to optimize the objective function by minimizing the sum of squared intra-cluster errors.

- **Hierarchical Clustering:** computes pairwise distance, and then merges similar clusters in a bottom-up fashion, without the need of providing the number of clusters. Usually, the result of hierarchical clustering comes as a dendrogram (Figure 3.4).

Clustering has been used in many application domains including biology, medicine, anthropology, marketing, and economics. It is also a vital process for condensing and summarizing information, since it can provide a synopsis of the stored data. Similar to query by content, there are two types of time series clustering: *whole clustering* and *subsequence clustering*.

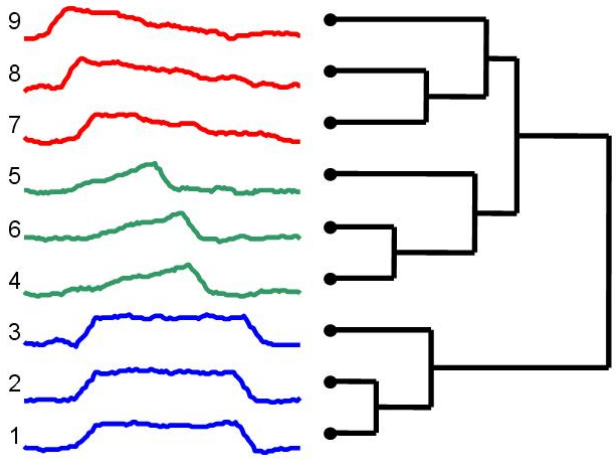

Figure 3.4. A hierarchical clustering of time series

Slika 3.4. Hijerarhijsko grupisanje vremenskih serija

3.2.4. Prediction (Forecasting)

Prediction can be viewed as a type of clustering or classification. The difference is that prediction is predicting a future state, rather than a current one. Its applications include forecasting of natural disasters (flooding, hurricane, snowstorm, etc), epidemics, stock crashes, etc. Many time series prediction applications can be seen in economic domains, where a prediction algorithm typically involves regression analysis. It uses known values of data to predict future values based on historical trends and statistics. Many techniques have been proposed to increase the accuracy of time series forecast, including the use of

neural network and dimensionality reduction techniques. More about prediction will be given in Chapter 4.

3.2.5. Summarization

Since time series data can be massively long, a summarization of the data may be useful and necessary. A statistic summarization of the data, such as the mean or other statistical properties can be easily computed even though it might not be particularly valuable or intuitive information. Instead, a *natural language*, *visualization* or *graphical summarization,* are often used to extract useful or meaningful information from the data. An Anomaly detection and a motif discovery are special cases of summarization where only irregular/repeating patterns are of interest. Summarization can also be viewed as a special type of clustering problem that maps data into subsets with associated simple (text or graphical) descriptions and provides a higher-level view of the data. This new simpler description of the data is then used in place of the entire dataset. The summarization may be done at multiple granularities and for different dimensions.

Some of popular approaches for visualizing massive time series datasets include *TimeSearcher* [Hochheiser and Shneiderman 2001], *Calendar-Based Visualization* [Wijk and Selow 1999], *Spiral* [Weber et al. 2000] and *VizTree* [Lin et al. 2004].

3.2.6. Anomaly Detection

In time series Data Mining and monitoring, the problem of detecting anomalous/ surprising/novel patterns has attracted much attention. In contrast to subsequence matching, anomaly detection is identification of previously unknown patterns. The problem is particularly difficult because what is considered as an anomaly can greatly differ depending on the task and/or domain. In a general sense, an anomalous behaviour is one that differs from "normal" behaviour. While there have been numerous definitions given for anomalous or surprising behaviours, the one given by [Keogh et al. 2002] is unique in that it requires no explicit formulation of what is anomalous. Instead, the authors simply define an anomalous pattern as "*pattern whose frequency of occurrences differs substantially from that expected, given previously seen data*". The problem of anomaly detection in time series has been generalized to include the detection of surprising or interesting

patterns (which are not necessarily anomalies). Anomaly detection is closely related to Summarization.

3.2.7. Segmentation

Segmentation in time series is often referred to as a dimensionality reduction algorithm. Although the segments created could be polynomials of an arbitrary degree, the most common representation of the segments is as linear functions. Probably the most intuitive segmentation method is a Piecewise Linear Representation (PLR). It refers to the approximation of a time series Q, of length n, with K straight lines, where $K<<n$.

Although appearing under different names and with slightly different implementation details, most time series segmentation algorithms can be grouped into one of the following three categories:

- **Sliding-Windows:** A segment is grown until it exceeds some error bound. The process repeats with the next data point not included in the newly approximated segment.
- **Top-Down:** The time series is recursively partitioned until some stopping criteria is met.
- **Bottom-Up:** Starting from the finest possible approximation, segments are merged until some stopping criteria are met.

The quality of a segmentation algorithm can be measured in several ways. The most obvious way is to measure the reconstruction error for a fixed number of segments. The reconstruction error is simply the Euclidean distance between the original data and the segmented representation.

3.3. TIME SERIES REPRESENTATIONS

As already mentioned, time series datasets are typically very large. For example, just eight hours of electroencephalogram data can require a few gigabytes of storage. Rather than analyzing or finding statistical properties on time series data, the goal of Data Mining tasks is changed to discovering useful information from the massive amount of data efficiently. This is a problem because for almost all Data Mining tasks, most of the

execution time spent by algorithm is used simply to move data from disk into main memory.

The straightforward method for indexing time series data is to consider a time series *C* of a length *n* as a point in a *n*-dimensional space. The interpretation of distances, described in the Section 3.1, can be seen in Figure 3.5. This approach suggests that time series could be indexed by Spatial Access Methods such as the R-tree [Guttman 1984]. However, most Spatial Access Methods begin to degrade rapidly at dimensionalities greater than 8-12, and realistic queries typically contain sometimes several hundreds or thousands datapoints. In order to utilize the Spatial Access Methods it is necessary to perform dimensionality reduction first. In [Faloutsos et al. 1994] the authors introduced GEneric Multimedia INdexIng method (GEMINI) which can exploit any dimensionality reduction method to allow efficient indexing.

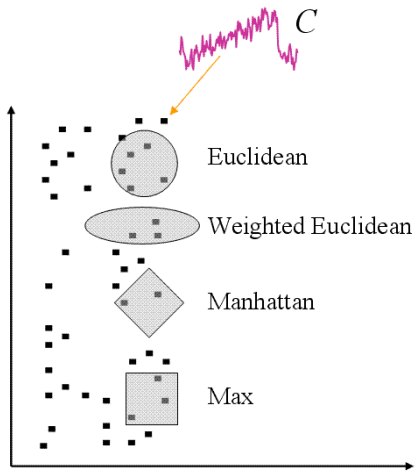

Figure 3.5. Time series as points in *n*-dimensional space and the interpretation of distances.

Slika 3.5. Vremenske serije kao tačke u n-dimenzionalnom prostoru i interpretacija rastojanja.

A crucial result in [Faloutsos et al. 1994] is that the authors proved that in order to guarantee no false dismissals, the distance measure in the index space must satisfy the following condition:

$$D_{index\_space}(A,B) \leq D_{true}(A,B) \qquad (3.6)$$

This theorem is known as the *lower bounding lemma*. Given the lower bounding lemma, and one of the Spatial Access Methods, GEMINI requires just the following three steps.

- Establish a distance metric from a domain expert (for example Euclidean distance).
- Produce a dimensionality reduction technique that reduces the dimensionality of the data from n to N $(N<<n)$, where N can be efficiently handled by chosen Spatial Access Method.
- Produce a distance measure defined on the N dimensional representation of the data, and prove that it obeys $D_{index\_space}(A,B) \leq D_{true}(A,B)$.

When the lower bounding lemma is satisfied for the defined distance measure, the algorithm for a ranged query is simple. For a query Q, and a distance ε the algorithm for finding all the neighbours of a Q in the range ε is:

- Project the query Q into the same feature space as the index.
- Find all candidate objects in the index within ε of the query.
- Retrieve from disk the actual sequences pointed to by the candidates.
- Compute the actual distances, and discard false alarms.

The efficiency of the GEMINI query algorithms depends on the quality of the transformation used to build the index. The tighter the bound on $D_{index\_space}(A,B) \leq D_{true}(A,B)$ the better. Ideally, the desired case would be $D_{index\_space}(A,B) = D_{true}(A,B)$. In that case, no false alarms would appear, but it is generally hard to realize. The more realistic situation is to allow a few false alarms, and discard them while computing the actual distance.

It may seem strange that, after all the efforts to collect and store the precise values of a time series, the exact values are discarded for some high level approximation. However, there are two important reasons why this is so:

- The users are typically not interested in the exact values of each time series data point. Rather, they are interested in the trends, shapes and patterns contained within the data. These may be captured best in some appropriate high-level representation.
- As a practical matter, the size of the database may be much larger than we can effectively deal with. In such instances, some transformation to a lower dimensionality representation of the data may allow more efficient storage and computation of the data.

No one representation of time series is superior for all tasks. There is even not clear how to choose the best representation for the chosen problem. For this reason, some of the well-known representations will be discussed here.

3.3.1. Discrete Fourier Transform

The first technique suggested for dimensionality reduction of time series was the Discrete Fourier Transform (DFT) [Agrawal et al. 1993]. The basic idea of a spectral decomposition is that any signal, no matter how complex, can be represented by the super position of a finite number of sine/cosine waves (Figure 3.6), where each wave is represented by a single complex number known as a Fourier coefficient. A time series represented in this way is said to be in the frequency domain. A signal of length *n* can be decomposed into *n/2* sine/cosine waves that can be recombined (without loss of information) into the original signal. However, many of the Fourier coefficients have very low amplitude and thus contribute little to reconstructed signal. These low amplitude coefficients can be discarded without much loss of information thereby saving storage space.

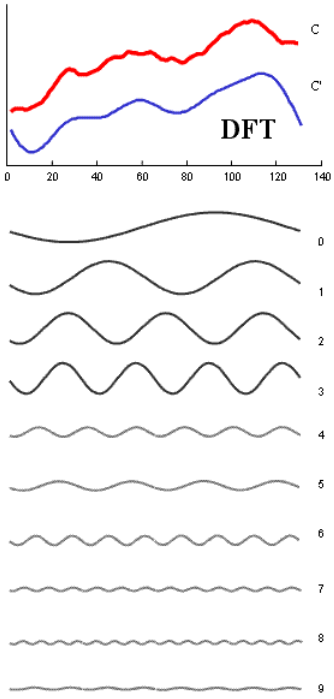

Figure 3.6. Discrete Fourier Transform as dimensionality reduction technique.

Slika 3.6. „Discrete Fourier Transform“ kao tehnika redukcije dimenzionalnosti.

To perform the dimensionality reduction of a time series *C* of length *n* into a reduced feature space of dimensionality *N*, the Discrete Fourier Transform of *C* is calculated. The transformed vector of coefficients is truncated at *N/2*. The reason the truncation takes place at *N/2* and not at *N* is that each coefficient is a complex number, and therefore we need one dimension each for the imaginary and real parts of the coefficients.

Given this technique to reduce the dimensionality of data from *n* to *N*, and the existence of the lower bounding distance measure, DFT can be easily integrated into the GEMINI framework. The time taken to build the entire index depends on the length of the queries for which the index is built. When the length is an integral power of two, an efficient algorithm can be employed (O*(mn*log*(n))*, where *n* is the length of one time series and *m* is the size of the database).

Although, DFT has many advantages (good ability to compress most natural signals, fast indexing algorithm etc.), it has also some disadvantages. The most important are:

- Difficult to deal with sequences of different lengths.
- Cannot support weighted distance measures.
- None of the implementations presented so far can guarantee no false dismissals

3.3.2. Discrete Wavelet Transform

Wavelets are mathematical functions that represent the data or other functions in terms of the sum and difference of a prototype function, so called the "analyzing" or "mother" wavelet. In this sense, they are similar to DFT. However, one important difference is that wavelets are localized in time, i.e. some of the wavelet coefficients represent small, local subsections of the data being studied. This is in contrast to Fourier coefficients that always represent global contribution to the data.

The first few coefficients contain an overall, coarse approximation of the data, while the later coefficients can be imagined as "zooming-in" to areas of high detail, as illustrated in Figure 3.7.

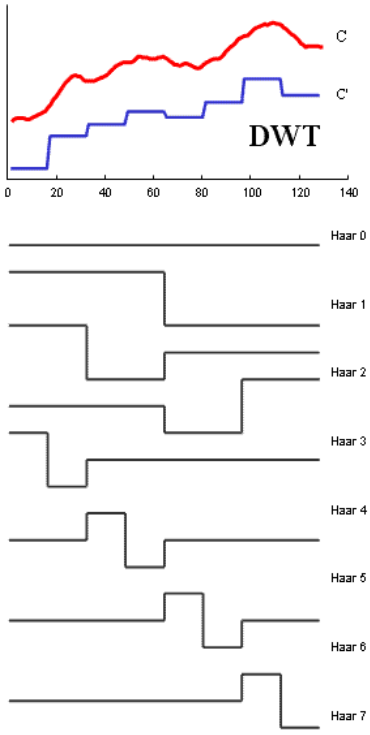

Figure 3.7. Discrete Wavelet Transform as dimensionality reduction technique.

Slika 3.7. „Discrete Wavelet Transform" kao tehnika redukcije dimenzionalnosti.

Chan and Fu [Chan and Fu 1999] produced a breakthrough for time series indexing with wavelets by producing a distance measure defined on wavelet coefficients which provably satisfies the lower bounding requirement. This contribution resulted in an explosion of interest in using wavelets for data compression, filtering, analysis, and other areas where the Fourier methods have previously been used. The work is based on a simple, but powerful type of wavelet known as the Haar Wavelet. The Discrete Haar Wavelet Transform (DWT) can be calculated efficiently and an entire dataset can be indexed in $O(mn)$.

The advantages of DWT include: good ability to compress most natural signals and fast indexing algorithm. Disadvantages of this technique are:

- Difficult to deal with sequences of different lengths (It is only defined for sequences whose length is an integral power of two).
- Cannot support weighted distance measures.
- None of distance measures using Haar wavelets are indexable.

3.3.3. Singular Value Decomposition

Singular Value Decomposition (SVD) has been initially successfully used for indexing images and other multimedia objects and recently has been proposed for time series indexing [Chan and Fu 1999]; [Korn et al. 1997].

Singular Value Decomposition is similar to DFT and DWT in that it represents the shape in terms of a linear combination of basis shapes (in this case they are called *eigenwaves*), as shown in Figure 3.8. However, SVD differs from DFT and DWT in one very important aspect DFT and DWT are local; they examine one data object at a time and apply a transformation. These transformations are completely independent of the rest of the data. In contrast, SVD is a global transformation. The entire dataset is examined and is then rotated such that the first axis has the maximum possible variance, the second axis has the maximum possible variance orthogonal to the first, the third axis has the maximum possible variance orthogonal to the first two, etc. The 2-dimensional case can be seen in Figure 3.9. This global characteristic of the transformation is both the weakness and the strength from an indexing point of view.

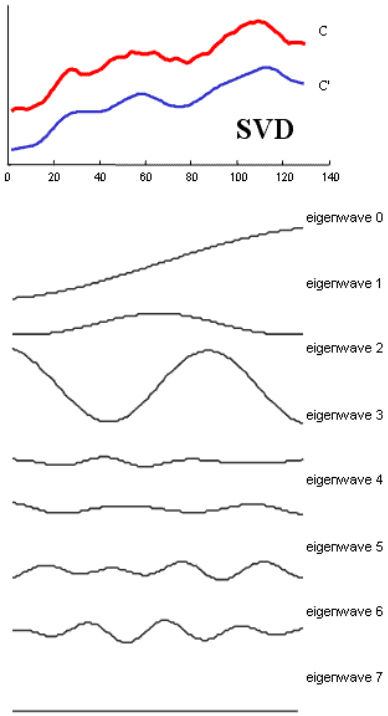

Figure 3.8. Singular Value Decomposition as dimensionality reduction technique.

Slika 3.8. „Singular Value Decomposition“ kao tehnika redukcije dimenzionalnosti.

SVD is the optimal transform in several senses. If we take the SVD of some dataset, and attempt to reconstruct the data, SVD is the optimal (linear) transform that minimizes reconstruction error. Also, the values of eigenwaves contain some information about the underlying structure of the all dataset. However, SVD has several drawbacks as an indexing scheme:

- Complexity. The classic algorithms to compute SVD require $O(mn^2)$ time and $O(mn)$ space.
- An insertion to the database requires recomputing the entire index.
- Cannot support weighted distance measures or non Euclidean measures

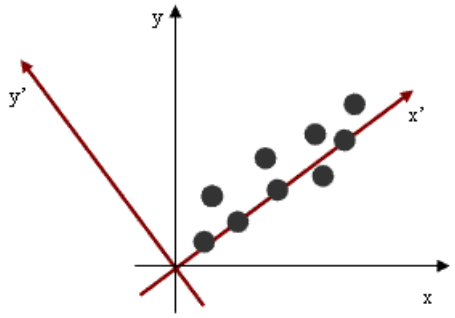

Figure 3.9. SVD algorithm in 2-dimensional space.

Slika 3.9. „Singular Value Decomposition“ algoritam u dvodimenzionalnom prostoru.

3.3.4. Piecewise Linear Approximation

The idea of using the piecewise linear segments to approximate time series dates back to 1970s [Pavlidis and Horowitz 1974]. This representation has numerous advantages, including data compression and noise filtering. Figure 3.10 shows an example of a time series represented by piecewise linear segments.

There are numerous algorithms available for segmenting time series. The complexity of *Top Down* algorithm is $O(n^2N)$ which is rather slow for most real-world applications, while the complexity of *Bottom-Up* and *Sliding window* algorithms is $O(n(1/C_{Ratio}))$ which is more acceptable ($C_{Ratio}=N/n$ is the compression ratio).

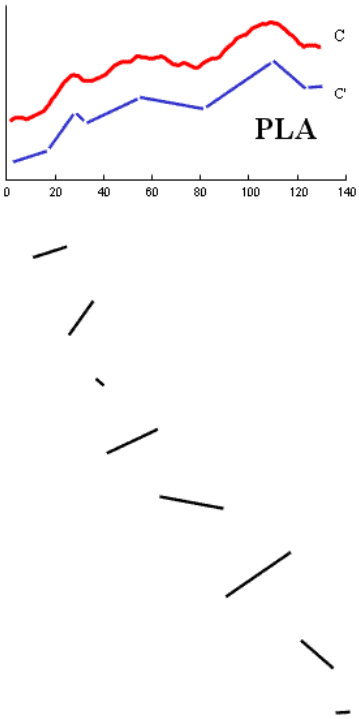

Figure 3.10. Piecewise Linear Approximation as dimensionality reduction technique.

Slika 3.10. „Piecewise Linear Approximation" kao tehnika redukcije dimenzionalnosti.

An open question is: how to best choose *K*, the "optimal" number of segments used to represent a particular time series. This problem involves a tradeoff between accuracy and compactness, and clearly has no general solution.

PLA is widely accepted in some communities (ie. biomedical) mainly because it's good characteristics: it has a good ability to compress natural signals, fast linear algorithm exists, supports weighted measures etc. The main disadvantage of PLA is that it isn't indexable by any data structure (but the sequential scan very fast).

3.3.5. Piecewise Aggregate Approximation

The basic idea in Piecewise Aggregate Approximation (PAA) is to approximate a time series by dividing it into equal-length segments and recording the mean value of the data points that fall within the segment [Keogh et al. 2001].

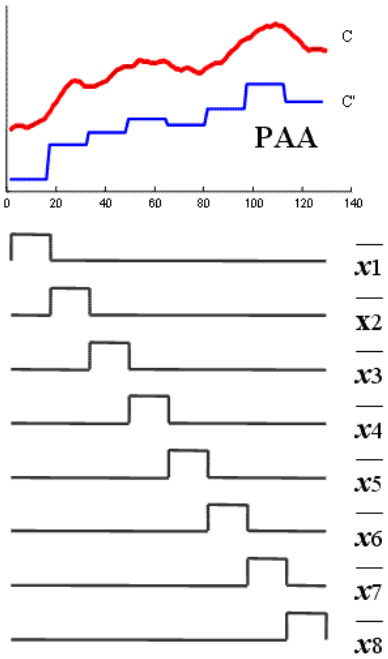

Figure 3.11. Piecewise Aggregate Approximation as dimensionality reduction technique.

Slika 3.11. „Piecewise Aggregate Approximation" kao tehnika redukcije dimenzionalnosti.

This representation reduces the data from n dimensions to N dimensions by dividing the time series into N equi-sized 'frames'. The mean value of the data in a frame is calculated, and a vector of these values becomes the data reduced representation. A time series C of length n is represented in N space by a vector $C'=(c'_1, \ldots, c'_N)$. The i^{th} element of C' is calculated by the following equation:

$$c'_i = \frac{N}{n} \sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i} c_j \qquad (3.7)$$

When $N = n$, the transformed representation is identical to the original representation. When $N = 1$, the transformed representation is simply the mean of the original sequence. More generally, the transformation produces a piecewise constant approximation of the original sequence, hence the name, the Piecewise Aggregate Approximation (PAA). This representation is also capable of handling queries of variable lengths.

PAA can be visualized as approximating a sequence with a linear combination of box functions. Figure 3.11 illustrates this idea.

PAA has recently been widely used mainly because of its numerous advantages. Some of them are: extremely fast to calculate, supports queries of arbitrary lengths, supports any Minkowski metric and non Euclidean measures, supports weighted Euclidean distance simple, intuitive, etc.

3.3.6. Adaptive Piecewise Constant Approximation

As an extension to the PAA representation, Adaptive Piecewise Constant Approximation (APCA) is introduced in [Keogh et al. 2001]. This representation allows the segments to have arbitrary lengths, which in turn needs two numbers per segment. The first number records the mean value of all the data points in a segment, and the second number records the length of the segment (Figure 3.12).

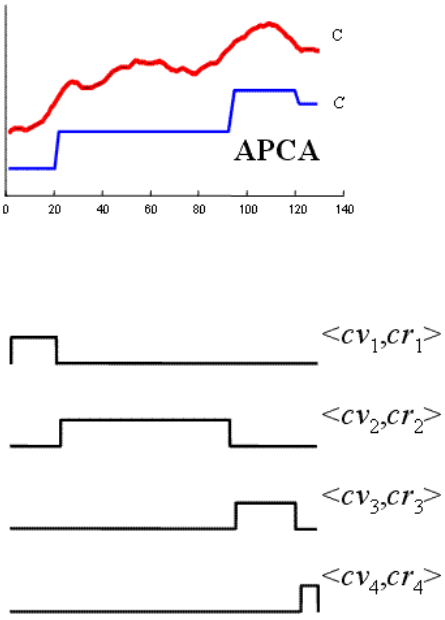

Figure 3.12. Adaptive Piecewise Constant Approximation as dimensionality reduction technique.

Slika 3.12. „Adaptive Piecewise Constant Approximation“ kao tehnika redukcije dimenzionalnosti.

It is difficult to make any intuitive guess about the relative performance of this technique. On one hand, PAA has the advantage of having twice as many approximating segments. On the other hand, APCA has the advantage of being able to place a single segment in an area of low activity and many segments in areas of high activity. In addition, one has to consider the structure of the data in question. It is possible to construct artificial datasets, where one approach has an arbitrarily large reconstruction error, while the other approach has reconstruction error of zero.

PAA and APCA have the same advantages and disadvantages, although APCA is a little bit harder to implement. PAA has proved better on some datasets (where all parts of time series are equally important), while APCA proved better on some other (where time series have little details in some places and high details in other places).

3.3.7. Symbolic Aggregate Approximation

Symbolic Aggregate Approximation (SAX) is a novel symbolic representation for time series introduced by [Lin et al. 2003].

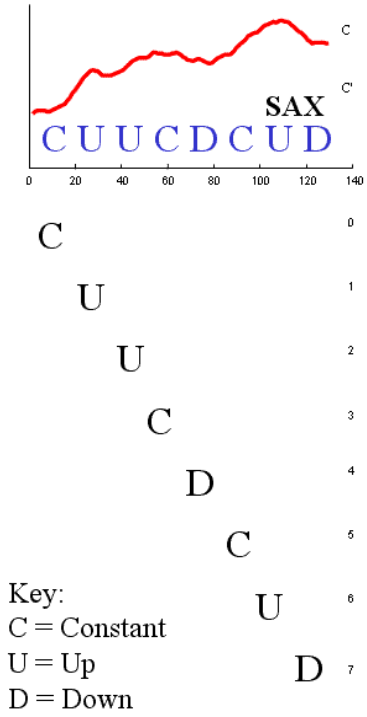

Figure 3.13. Symbolic Aggregate Approximation as dimensionality reduction technique.

Figure 3.13. Symbolic Aggregate Approximation as dimensionality reduction technique.

The basic idea of SAX is to convert the time series into an alphabet of discrete symbols and then to use string indexing techniques to manage the data. A simple example is shown in Figure 3.13, where the alphabet consists of just 3 symbols with the meaning: **C**-constant, **U**-Up, **D**-Down.

This is a potentially interesting idea mainly because techniques from the well developed field of string processing can be used. However, many of the aspects are still not clear: how to discretize time series, how big should be the alphabet, how to support the Euclidean distance etc.

Chapter IV

TIME-SERIES FORECASTING

Prediction, as a philosophical concept, can be defined as a statement that a certain kind of event will occur at some future time [Turchin et al. 2006]. In regards to predicting the future Howard H. Stevenson says: *Prediction is at least two things: Important and hard. Important, because we have to act, and hard because we have to realize the future we want, and what is the best way to get there.*

What distinguishes prediction in science from its common usage is the fact, that it must be an explicit scientific theory on which the prediction is based. This requirement clearly distinguishes the prediction in science from prophecies, astrological predictions or some other non-scientific predictions. Within the scientific usage three kinds of predictions can be further distinguished [Turchin et al. 2006]:

- **Projection:** Conceptually the simplest kind of a prediction. The problem is to answer "what if" question: assuming certain initial conditions and a certain mechanism of change, what would be the future trajectory of the modelled system?
- **Forecast:** A forecast is a prediction that a certain variable will reach a specified level (or will be within the specified range of values) at a certain point in future. Unlike the projection, forecasting requires that the validity of the assumptions of the underlying theory must be accepted.

- **Scientific prediction:** Scientific prediction is used to test scientific theories. Scientific prediction inverses the logic of forecasting: whereas in making forecasts we assume the validity of the underlying theory and want to know what will happen to observed features, in a scientific prediction the observed features are used to deduce the validity of the theory.

The Forecast is the most common (and probably the only one) prediction method used for time series data. The main reason for choosing the forecast method is because it deals with numerical data of which always consists time series. Furthermore, the projection is not used because it would be too trivial and useless in practice. The Scientific prediction could not be applied to time series because the knowledge about underlying theory which describes time series behaviour is usually weak. Moreover, it is sometimes unclear if any kind of such theory even exists.

In time series forecasting we assume to know nothing about the causality that affects the variable we are trying to forecast [Arsham-Web]. Instead, the past behavior of time series is examined in order to infer something about its future behavior. The method used to produce a forecast may involve the use of a simple deterministic model such as a linear extrapolation or the use of a complex stochastic model for adaptive forecasting.

One example of the use of time-series analysis would be the simple extrapolation of a past trend in predicting population growth. Another example would be the development of a complex linear stochastic model for passenger loads on an airline. Time-series models have been used to forecast the demand for airline capacity, seasonal telephone demand, the movement of short-term interest rates, and other variables from different domains. Time-series models are particularly useful when little is known about the underlying process one is trying to forecast.

In the Single-Equation Regression Models the variable under study is explained by a single function (linear or nonlinear) of a number of explanatory variables. The equation will often be time-dependent (i.e., the time index will appear explicitly in the model), so that one can predict the response over time of the variable under study to changes in one or more of the explanatory variables. A principal purpose for constructing single-equation regression models is forecasting. The forecast is a quantitative estimate (or set of estimates) about the likelihood of future events which is developed on the basis of

past and current information. This information is embodied in the form of a model. By extrapolating these models beyond the period over which they were estimated, we can make forecasts about near future events.

The choice of the type of a model to develop involves trade-offs between time, energy, available resources, costs, and desired forecast precision. The construction of a multi-equation simulation model may require large expenditures of resources, time and money. The gains from this effort may include a better understanding of the relationships and structure involved as well as the ability to make a better forecast. However, in some cases these gains may be small enough to be outweighed by the heavy costs involved. Because the multi-equation model requires a great knowledge about the process being studied, the construction of such models may be extremely difficult.

The decision to build a time-series model usually occurs when little or nothing is known about the determinants of the variable being studied, when a large number of data points are available, and when the model is to be used largely for short-term forecasting. Given some information about the processes involved, however, it may be reasonable to construct both types of models and compare their relative performance.

Given the forecast problem, two types of forecasts can be distinguished:

- **Point forecasts** predict a single number in each forecast period,
- **Interval forecasts** indicate an interval in which we hope the realized value will lie.

The information provided by the forecasting process can be used in many ways. An important concern in forecasting is the problem of evaluating the nature of the forecast error by using appropriate statistical tests. The best forecast is the one which yields the forecast error with the minimum variance. In the single-equation regression model, an ordinary lest-squares estimation yields the best forecast among all linear estimators having minimum mean-square error [Arsham-Web].

The error associated with a forecasting procedure can come from a combination of four distinct sources:

- The random nature of the additive error process in a linear regression model guarantees that forecasts will deviate from true values even if the model is specified correctly and its parameter values are known
- The process of estimating the regression parameters introduces error because estimated parameter values are random variables that may deviate from the true parameter values.
- In the case of a conditional forecast, errors are introduced when forecasts are made for the values of the explanatory variables for the period in which the forecast is made.
- Errors may be introduced because the model specification may not be an accurate representation of the "true" model.

As time series has been investigated for a long time, many of the forecasting methods were developed. They range from very simple and intuitive methods such as Linear prediction or Moving average method to very complex stochastic modelling methods such as ARMA models or Box-Jenkins method. Among many of the time series forecasting methods three main groups can be identified [Wikipedia]:

- Standard time series methods,
- Causal/Econometric methods, and
- Data mining methods.

Time series forecasting is the most important real-world application of the system presented in this thesis. For that reason, this chapter will give the overview of the state of the art in this area. In the rest of this chapter, three main groups of forecasting methods and their most common particular methods will be explained.

4.1. STANDARD TIME SERIES METHODS

Standard time series forecasting methods have chronologically appeared first. They are characterized by relatively simple mathematical modelling techniques which mainly resulted from some other time series problems such as smoothing or interpolation. These methods are widely accepted in many different areas, which is probably the consequence of their simplicity and robustness. Some of the most common standard time series forecasting methods are:

- Moving average
- Exponential smoothing
- Extrapolation
- Linear prediction
- Trend estimation

4.1.1. Moving Average

Moving average method was originally used as a smoothing technique and later it was used for forecasting. Smoothing techniques are used to reduce irregularities (random fluctuations) in time series data. They provide a clearer view of a true underlying behaviour of the series. Moving average ranks among the most popular techniques for the pre-processing of time series. It is used to filter random "white noise" from the data, to make the time series smoother or even to highlight certain informational components contained in the time series [Arsham-Web].

Moving average is a statistical technique used to analyze a set of data points by creating an average of one subset of the full data set at a time with each number in the subset given an equal statistical weight. So the moving average is not a single number, but it is a set of numbers, each of which is the average of the corresponding subset of a larger set of data points.

There are several important variations of the moving average technique [Moving Averages page]. *Prior moving average of order n* calculates the unweighted mean of the previous *n* data points. For the time series $C= (c_1, c_2, \ldots, c_p)$ it is calculated according to equation:

$$MA_{t+1} = \frac{c_t + c_{t-1} + \ldots + c_{t-n+1}}{n} \qquad (4.1)$$

The parameter *n* defines the level of abstraction that is needed for smoothing. Higher levels of *n* give smoother curve, while lower levels of *n* could contain more information

about small fluctuations. Figure 4.1 illustrates the choice of the parameter *n*. For the same time series the MA(2) and MA(5) are shown on figures a) and b) respectively.

a) b)

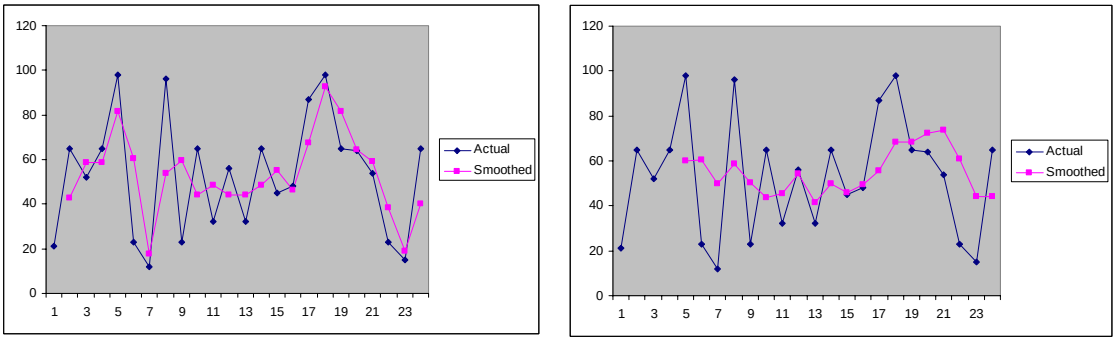

Figure 4.1. The MA(2) and MA(5) for the same time series

Slika 4.1. MA(2) i MA(5) za istu vremensku seriju.

Central moving average is a modification of the prior moving average where the value in time *t* is calculated by using *n* previous and *n* subsequent values:

$$CMA_t = \frac{c_{t-n} + c_{t-n+1} + \ldots + c_{t-1} + c_t + c_{t+1} + \ldots + c_{t+n-1} + c_{t+n}}{2n+1} \qquad (4.2)$$

Weighted moving average assigns different weights to different data points. Sometimes, it is very convenient to assign higher weights to more recent points and lower weights to points which are "older". Weighted moving average of order *n* is defined:

$$WMA_{t+1} = w_1 c_t + w_2 c_{t-1} + \ldots + w_n c_{t-n+1} \qquad (4.3)$$

where $w_1+w_2+\ldots+w_n=1$.

There are also some other variations of the moving average such as *Cumulative moving average* or *Exponential moving average* but they are, more or less, modifications of the mentioned ones.

Moving average methods (with exception of Central MA) can be used for short-term forecasts. When the values until time *t* are known these methods can predict the value in the time *t+1*, as can be seen in equations (4.1) and (4.3). It is also possible to include the predicted value (*t+1*) in the known values and then to predict *t+2* value etc., but this is rarely done because the errors are accumulated.

4.1.2. Exponential Smoothing

Exponential smoothing is a very popular scheme either to produce a smoothed time series or to make forecast. It is similar to Weighted moving average with the exception that Exponential Smoothing assigns exponentially decreasing weights as the observation get older. In other words, recent observations are given relatively more weight in forecasting than the older observations.

For the time series $C= (c_1, c_2, \ldots, c_p)$ the equation for calculation of the Exponential smoothing is [Arsham-Web]:

$$F_{t+1} = \alpha c_t + (1-\alpha)F_t \qquad (4.4)$$

where F_t is the forecasted value of time series, and α is the weighting factor which ranges from 0 to 1.

Values of α close to one have less of a smoothing effect and give greater weight to recent changes in the data, while values of α closer to zero have a greater smoothing effect and are less responsive to recent changes. This is illustrated in Figure 4.2 where on the same time series a Smoothing average was applied with the α=0.2 (figure a) and α=0.8 (figure b).

There is no formally correct procedure for choosing α. Sometimes the statistician's judgment is used to choose an appropriate factor. Alternatively, a statistical technique may be used to optimize the value of α. For example, the method of least squares might be used to determine the value of α for which the sum of the quantities $(F_{n-1} - c_n)^2$ is minimized [Wikipedia].

a) b)

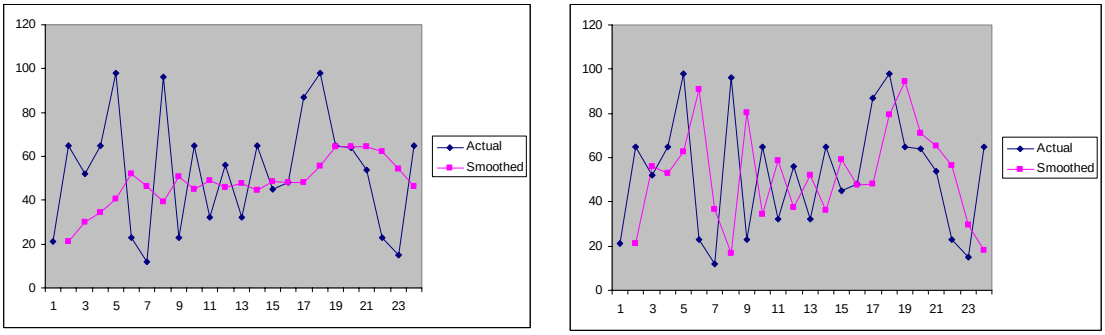

Figure 4.2. Smoothing average with α=0.2 (figure a) and α=0.8 (figure b)

Slika 4.2. „Smoothing average“ za α=0.2 (slika a) i α=0.8 (slika b)

Similarly as the Moving average, Exponential smoothing can be used for the short-term forecasts. When the *t* values are known, the *t+1* value can be forecasted by using the equation (4.4). The technique of including the *t+1* forecast in the known values in order to forecast *t+2* value can also be applied here.

4.1.3. Extrapolation

Extrapolation is the process of constructing new data points outside a discrete set of known data points [Brezinski and Redivo Zaglia 1991]. It is similar to the process of interpolation, which constructs new points between known points, but the results of extrapolations are often less accurate, and must be accepted with greater uncertainty.

The choice of which extrapolation method to apply relies on a prior knowledge of the process that created the existing data points. Crucial questions are for example if the data can be assumed to be continuous, smooth, periodic etc.

Any of the interpolation formulas can be used for the purpose of extrapolation. For example, *Linear extrapolation* denotes the act of creating a tangent line at the end of the known data and extending it beyond that limit. Linear extrapolation will only provide good results when used to extend the graph of an approximately linear function or not too far beyond the known data.

If the two data points, nearest to the point x_* to be extrapolated, are (x_{k-1},y_{k-1}) and (x_k,y_k), linear extrapolation gives the function:

$$y(x_*) = y_{k-1} + \frac{x_* - x_{k-1}}{x_k - x_{k-1}}(y_k - y_{k-1}) \qquad (4.5)$$

It is possible to include more than two points, and averaging the slope of the linear interpolant, by regression-like techniques, on the data points chosen to be included.

Polynomial extrapolation can be realized by creating a polynomial curve through the entire known data or just near the end. The resulting curve can then be extended beyond the end of the known data. Polynomial extrapolation is typically done by: Lagrange interpolation, Newton's method of finite differences or by using the splines. The resulting polynomial may be used to extrapolate the data.

Although the extrapolation methods are quite convenient the quality of a particular method of extrapolation is limited by the assumptions about the function made by the method. The main bottleneck, which often cannot be removed, is if the functional forms assumed by the chosen extrapolation method accurately represent the nature of the function being extrapolated.

4.1.4. Linear Prediction

Linear prediction is a method for signal source modelling dominant in speech signal processing and having wide application in other areas [Linear prediction Web]. It is a mathematical operation where future values of a time series are estimated as a linear function of previous samples.

For the time series $C= (c_1, c_2, \ldots, c_p)$ the most common representation of Linear prediction is:

$$c'_n = -\sum_{i=1}^{j} a_i c_{n-i} \qquad (4.6)$$

where c'_n is the predicted value of the time series C and a_i are predictor coefficients.

There are different methods for the computation of the predictor coefficients but the most common is probably an *Autocorrelation Method*. The main idea of this method is to minimize the sum of squared errors in order to calculate predictor coefficients (a_i):

$$\sum_n (c'_n - c_n)^2 \qquad (4.7)$$

Linear prediction is a very robust technique and is widely accepted in audio signal processing and speech processing for representing a digital signal of speech in compressed form. However, in forecasting it can be used only for short-term forecasts for predicting only the next value of time series.

4.1.5. Trend estimation

Trend estimation is the application of statistical techniques to make and justify statements about trends in the data [Bianchi et al. 1999]. By assuming that the underlying process is a physical system that is incompletely understood, one may construct a model, independent of anything known about the physics of the process, to explain the behaviour of a measurement. In particular, one may wish to know if the measurements show an increasing or decreasing trend, which can be statistically distinguished from a random behaviour.

The model, in this case, is a function fitted through the data. Given a set of data, and the intention to produce a "model" of that data, there are a variety of functions that can be chosen for the fit. But if there is no prior understanding of the data, the simplest function to fit is a straight line.

Far most common method for fitting a straight line through the set of data is the *least squares* method. The main idea of least squares method is as follows: given a set of data points t_i, and data values c_i, the goal is to choose coefficients a and b so that the following expression is minimized:

$$\sum (((at_i + b) - c_i)^2) \qquad (4.8)$$

When the coefficients *a* and *b* are known a time series with the trend can be represented in the following manner:

$$c_i = at_i + b + \varepsilon_i \qquad (4.8)$$

where ε_i are independent randomly distributed "errors". Unless something special is known about the ε's, they are assumed to have a normal distribution.

Trend estimation shows good forecasting results in some applications. For example, the measurement of a global surface temperature recorded in the past 140 years (http://www.ipcc.ch/) showed a stable trend of about 0.6ºC per year with the variation of 0.2ºC [Wikipedia]. However, the trend estimation is too simple for most real-world applications since most of the times series does not have constant, stable trends and sometimes even normal distribution of errors.

4.2. CAUSAL/ECONOMETRIC METHODS

These methods try to exceed all the bottlenecks of the standard forecasting methods. The main idea, which lies behind the Causal methods, is to (mathematically) model a process which generated a time series, and to use a gained model for prediction. The model is usually generated by using statistical modelling or regression analysis.

These methods are commonly used in financial prediction mainly because of its capabilities to model a wide range of processes. Although, the Causal/Econometric methods are designed for short-term forecasts, they can be reasonably good applied for middle-term forecasts. Some of the most frequently used methods from this group are:

- Regression analysis
- Autoregressive moving average (ARMA)
- Box-Jenkins methodology

4.2.1. Regression Analysis

Regression analysis is a collective name for techniques for the modelling and analysis of numerical data consisting of values of a *dependent variable* (also called response variable or

measurement) and of one or more *independent variables* (also known as explanatory variables or predictors) [Sykes Web]. The dependent variable in the regression equation is modelled as a function of the independent variables, corresponding parameters ("constants"), and an error part. The error part is treated as a random variable. It represents unexplained variation in the dependent variable [Han and Kamber 2006], [Weisberg 2005]. The parameters are estimated so as to give the "best fit" of the data. Most commonly the best fit is evaluated by using the least squares method, but other criteria have also been used.

Generally, regression equation is given in the following form:

$$Y = f(X, \beta) \qquad (4.9)$$

where the unknown parameter(s) is denoted as β (may be a scalar or a vector of length k), independent variables are denoted as X and dependent variable as Y.

The user of the regression analysis must make an intelligent guess about this function. Sometimes the form of this function is known, but sometimes it is not.

If N is the number of independent variables (the length of a vector X), two kinds of regression can be distinguished: if $N=1$ it represents a *simple regression* while when $N>1$ it represents *multiple regression*. Furthermore, if k is the length of a vector β three more cases can be distinguished:

- if $N < k$, regression analysis cannot be performed because there is not provided enough information to do so.
- if $N = k$, then the problem reduces to solving a set of N equations with N unknowns β.
- if $N > k$ regression analysis can be performed. Such a system is also called an *overdetermined system*.

According to the form of the regression function two main types of the regression can be distinguished:

- **Linear regression**. The model function is a linear combination of one or more model parameters, called regression coefficients. However, the function needs not to be linear in the independent variables.

- **Nonlinear regression**. The model function is not linear in the parameters. In this case, the data are fitted by a method of successive approximations which introduces many complications.

Regression can be used for forecasting, inference, hypothesis testing, and modelling of causal relationships. However, regression analysis has been criticized as being misused for these purposes in many cases where the appropriate assumptions cannot be verified to hold. One factor contributing to the misuse of regression is that it can take considerably more skill to critique a model than to fit a model.

4.2.2. Autoregressive Moving Average (ARMA)

Given a time series of data X_t, the ARMA model is a tool for understanding and, perhaps, predicting future values in this series. The model consists of two parts, an autoregressive (AR) part and a moving average (MA) part. The model is usually then referred to as the ARMA(p,q) model where p is the order of the autoregressive part and q is the order of the moving average part [Hamilton 1994].

The notation AR(p) refers to the autoregressive model of order p. In the autoregressive model, the next successive point is modelled as a linear combination of p previous points with an addition of some kind of uncertainty. The AR(p) model is given by:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t \qquad (4.10)$$

where φ_i are the parameters of the model, c is a constant and ε_t is a white noise. The white noise can be defined as a signal or array of independent identically-distributed random variables sampled from a normal distribution with zero mean: $\varepsilon_t \sim N(0,\sigma^2)$ where σ^2 is the variance.

The notation MA(q) refers to the moving average model of order q. In the moving average model, the next successive point is modelled as a mean of the time series with addition of linear combination of previous error terms and a current error term. The MA(q) model is given by:

$$X_t = \mu + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} + \varepsilon_t \qquad (4.11)$$

where μ is the mean of the time series X_t, θ_i are the parameters of the model and ε_t is a white noise error terms. Frequently, the mean of the time series μ is omitted for simplicity, because every time series can be transformed to a time series with a zero mean by some simple pre-processing steps.

The notation ARMA(p, q) refers to the model with p autoregressive terms and q moving average terms [Box and Jenkins 1970]. This model contains the AR(p) and MA(q) models:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \qquad (4.12)$$

The ARMA(p, q) model is defined with its orders p and q, and the parameters φ_i and θ_i. It is generally considered good practice to find the smallest values of p and q which provide an acceptable fit to the data. The parameters φ_i and θ_i can, after choosing p and q, be fitted by least squares regression to find the values of the parameters which minimize the error term. The more robust method for the model estimation is given by Box and Jenkins [Box and Jenkins 1970].

4.2.3. Box-Jenkins Methodology

Box-Jenkins modelling involves identifying an appropriate ARMA or ARIMA (Autoregressive integrated moving average) process, fitting it to the data, and then using the fitted model for forecasting. One of the attractive features of the Box-Jenkins approach to forecasting is that ARMA and ARIMA processes are a very rich class of possible models and it is usually possible to find a process which provides an adequate description to the data.

The original Box-Jenkins modelling procedure involved an iterative three-stage process of *model selection*, *parameter estimation* and *model checking*. Recent explanations of the process [Makridakis et al. 1998] often add a preliminary stage of *data preparation* and a final stage of *model application (or forecasting)*.

Data preparation involves transformations and differencing. Transformations of the data (such as square roots or logarithms) can help to stabilize the variance in a series where the variation changes with the level. Then the data are differenced until there are no obvious patterns such as a trend or seasonality left in the data. Differencing, in this case, means taking the difference between consecutive observations. The differenced data are often easier to model than the original data.

Model selection in the Box-Jenkins framework uses various graphs based on the transformed and differenced data to try to identify potential ARMA or ARIMA processes which might provide a good fit to the data. Usually, the plots of the autocorrelation and partial autocorrelation functions of time series are used in order to decide which (if any) autoregressive or moving average component should be used in the model.

Parameter estimation means finding the values of the model coefficients which provide the best fit to the data. The most common methods use *maximum likelihood estimation* or *non-linear least-squares estimation*.

Model checking involves testing of the assumptions of the model to identify any areas where the model is inadequate. If the model is found to be inadequate, it is necessary to go back to Step 2 and try to identify a better model.

Forecasting is what the whole procedure is designed to accomplish. Once the model has been selected, estimated and checked, it is usually a straight forward task to compute forecasts.

Although originally designed for modelling time series with ARMA and ARIMA processes, the underlying strategy of the Box and Jenkins is applicable to a wide variety of statistical modelling situations. It provides a convenient framework which allows an analyst to think about the data, and to find an appropriate statistical model which can be used to help answer relevant questions about the data.

4.3. DATA MINING METHODS

These methods are relatively novel if compared to the previously described methods. Also, the whole concept of these forecasting methods is quite different from the previous model driven methods. Data mining methods do not require the information about the model of underlying process which generated the time series. Instead, they learn from the past examples and implicitly create the model of the process which is used for forecasting. Furthermore, there are several distinguishing features of the data mining methods that make them valuable for a forecasting task.

As opposed to the traditional model based methods, the data mining methods are data-driven self-adaptive methods in that there are few a priori assumptions about the models for problems under study. They learn from examples and capture functional relationships among the data even if the underlying relationships are unknown or hard to describe. Thus data mining methods are well suited for problems whose solutions require knowledge that is difficult to specify but for which there are enough data or observations. These modelling approaches with the ability to learn from experience are very useful for many practical problems since it is often easier to have data than to have good theoretical guesses about the underlying laws of the systems from which the data are generated.

Furthermore, data mining methods can generalize. After learning the data presented to them (a sample), data mining methods can often correctly complete the unseen part of a problem even if the sample data contain noisy information.

In data mining, the prediction can be understood as the classification where the class is not categorical (discrete and not ordered) but continuous-valued (ordered) [Han and Kamber 2006]. However, among many classification data mining techniques only a few can be used for time series forecasting:

- Artificial neural networks
- Support vector machines
- k-nearest-neighbour technique
- Case-based reasoning.

4.3.1. Artificial Neural Networks

An artificial neural network (ANN), often just called a "neural network" (NN), is a mathematical model or computational model based on biological neural networks. They are composed of a number of interconnected simple processing elements called neurons or nodes. Each node receives an input signal which is the total "information" from other nodes or external stimuli, processes it locally through an activation or transfer function and produces a transformed output signal to other nodes or external outputs. This information processing characteristic makes ANNs a powerful computational methodology: they are able to learn from examples and then to generalize to examples never seen before [Zhang et al. 1998].

Many different ANN models have been proposed since 1980s. Probably the most influential model is the multi-layer perceptrons (MLP). The MLP is typically composed of several layers of nodes. The first or the lowest layer is an input layer where external information is received. The last or the highest layer is an output layer where the problem solution is obtained. The input layer and the output layer are separated by one or more intermediate layers called the hidden layers. The nodes in neighbouring layers are usually fully connected by acyclic arcs from a lower layer to a higher layer. Figure 4.3 gives an example of a fully connected MLP with one hidden layer.

For an explanatory or causal forecasting problem, the inputs to the ANN are usually independent on one predictor variables. The functional relationship estimated by the ANN can be written as:

$$y = f(x_1, x_2, ..., x_p) \qquad (4.13)$$

where $x_1, x_2, \ldots, x_p$ are p independent variables and y is a dependent variable. In this sense, the neural network is functionally equivalent to a nonlinear regression model.

On the other hand, for an extrapolative or time series forecasting problem, the inputs are typically the past observations of the data series and the output is the future value. The ANN performs the following function mapping:

$$y_{t+1} = f(y_t, y_{t-1}, ..., y_{t-p}) \qquad (4.14)$$

where y_t is the observation at time t. Thus the ANN is equivalent to the nonlinear autoregressive model for time series forecasting problems.

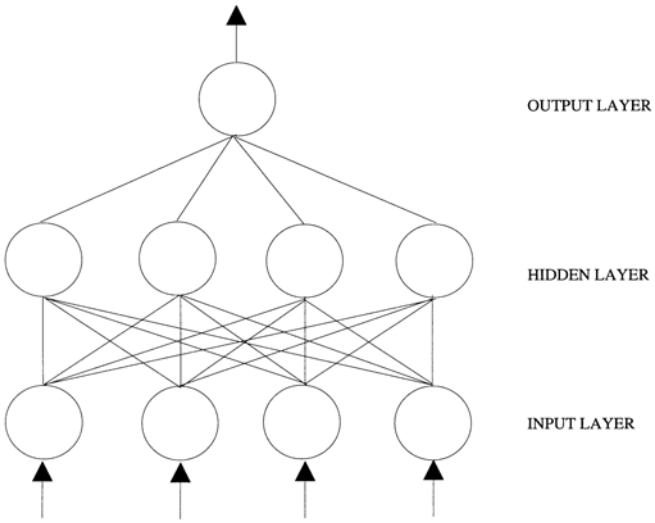

Figure 4.3. MLP with one hidden layer

Slika 4.3. MLP sa jednim skrivenim slojem

Before any ANN can be used to perform any desired task, it must be trained to do so. Basically, training is the process of determining the arc weights which are the key elements of the ANN. The knowledge learned by a network is stored in the arcs and nodes in the form of the arc weights and node biases. Mentioned knowledge containers of an ANN can carry out complex nonlinear mappings from its input nodes to its output nodes.

The methodology and organisation of the ANNs have several more advantages in time series forecasting problem:

- ANNs are universal functional approximators. It has been shown that a network can approximate any continuous function to any desired accuracy [Hornik et al. 1989]. ANNs have more general and flexible functional forms than the traditional statistical methods can effectively deal with.
- ANNs are nonlinear. The traditional approaches to time series prediction, such as the Box-Jenkins or ARIMA method assume that the time series under study are generated from linear processes. However, many of the real-world problems are nonlinear. ANNs are capable of performing nonlinear modelling without a priori knowledge about the relationships between the input and output variables. Thus they are a more general and flexible modelling tool for forecasting.

ANNs are probably the most common data mining approach for time series forecasting. Hence, they have been used in many disciplines including: forecasting of sunspot series [Li et al. 1990] [De Groot and Wurtz 1991], financial forecasting [Refenes 1995] [Gately 1996], electric load consumption forecasting [Bacha and Meyer 1992] [Srinivasan et al. 1994], environmental temperature forecasting [Balestrino et al. 1994], ozone level forecasting [Ruiz-Suarez et al. 1995] etc.

4.3.2. Support Vector Machines

Support vector machines (SVMs) are a set of related supervised learning methods used for classification and forecasting. Viewing input data as two sets of vectors in a *p*-dimensional space, the SVM will construct a separating hyperplane in that space, one which maximizes the margin between the two data sets. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the neighbouring datapoints of both classes, since in general – the larger the margin – the better the generalization error of the classifier. If such a hyperplane exists, it is clearly of interest and is known as the *maximum-margin hyperplane* and such a linear classifier is known as a *maximum margin classifier*.

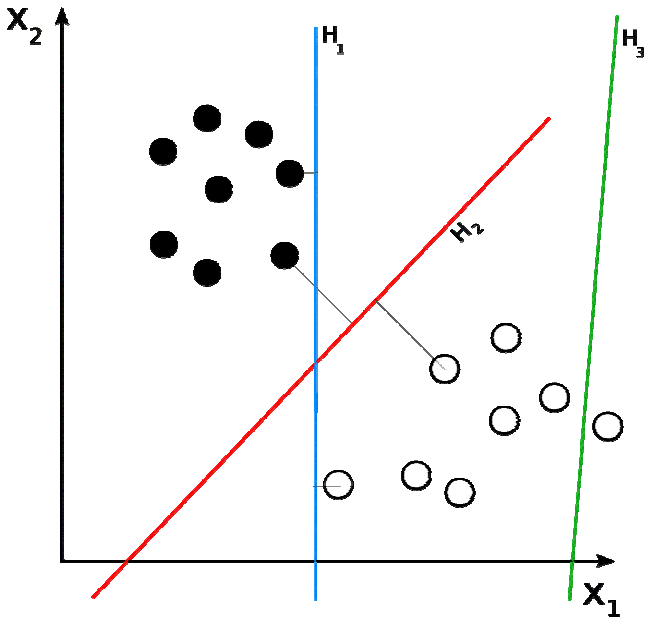

Figure 4.4. SVM separation of data points

Slika 4.4. SVM razdvajanje tačaka

Figure 4.4 illustrates the idea of SVMs. H3 (green) doesn't separate the 2 classes. H1 (blue) does, with a small margin and H2 (red) with the maximum margin.

The original optimal hyperplane algorithm was a linear classifier. However, many of the real world data are not linearly separable. A modification of the algorithm [Boser et al. 1992] suggested a way to create non-linear classifiers by applying a kernel trick to the maximum-margin hyperplanes. The idea of this modification is to transform the original data into a high dimensional space where the data are linearly separable and to fit the maximum-margin hyperplane in the transformed feature space. The transformation may be non-linear and the transformed space could be "very" high dimensional but the goal is achieved: the classifier is a hyperplane in the high-dimensional feature space (it may be non-linear in the original input space).

Forecasting by using the SVM is very similar. The basic idea is to map the data into a high dimensional feature space via a nonlinear mapping and to do a linear regression in this space [Müller et al. 1997]. Thus linear regression in a high dimensional feature space corresponds to the nonlinear regression in the low dimensional input space R^p.

SVMs are recently very often used in financial time series forecasting [Cao and Tay 2001] [Tay and Cao 2001] [Tay and Cao 2002] [Huang et al. 2005]. Time series of length p are considered as p-dimensional vectors and the linear regression is applied in a transformed space as already described. Some authors compared SVMs with neural networks and highlight several advantages:

- SVMs need a smaller number of free parameters compared to neural networks
- SVMs forecast better than neural networks [Cao and Tay 2001] according to several error measurements
- Training SVMs is faster than neural networks
- The BP network may not converge to global solutions while the solution of the SVMs is unique, optimal and global.

4.3.3. *k*-Nearest Neighbour Technique

k-nearest neighbour algorithm (*k*NN) is a method for classifying objects based on the closest training examples in the feature space [Wikipedia]. *k*NN is a type of instance-

based learning, or lazy learning, where the function is approximated only locally and all computation is postponed until classification. It can also be used for forecasting.

According to this technique, an object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours. The k is a positive integer, typically small. If $k = 1$, then the object is simply assigned to the class of its nearest neighbour. In binary (two class) classification problems, it is helpful to choose k to be an odd number as this avoids tied votes.

The neighbours are taken from a set of objects for which the correct classification is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. In order to identify neighbours, the objects are represented by position vectors in a multidimensional feature space. It is usual to use the Euclidean distance, though other distance measures, such as the Manhattan distance could in principle be used instead. The k-nearest neighbour algorithm is sensitive to the local structure of the data.

The same method can be used for forecasting. The prediction is based on the target outputs (predicted values) of the k nearest neighbours of the given query point [Hastie et al. 2001]. Specifically, given a data point we compute the Euclidean distance between that point and all points in the training set. Then we pick the closest k training data points and set the prediction as the average of the target output values for these k points. Speaking More formally, let $J(x)$ be the set of k nearest neighbours of point x. Then the prediction is given by [Nesreen et al. 2009]:

$$\hat{y} = \frac{1}{k} \sum_{m \in J(x)} y_m \qquad (4.15)$$

where y_m is a target output for training data point x_m.

Naturally k is a key parameter in this method, and has to be selected with care. A large k will lead to a smoother fit, and a lower variance, and vice versa for a small k. Sometimes, it can be useful to weight the contributions of the neighbours, so that the nearer neighbours contribute more to the average than the more distant ones [Plummer 2001].

4.3.4. Case-Based Reasoning

The foundations of Case-Based Reasoning technique are described in Chapter 2 in detail. Some of the results and applications of CBR systems in time series forecasting will be described here.

Although, it is a promising data mining technique, CBR was rarely used for time series forecasting. Probably the main reason for this is that the use of case-based reasoning for time series processing introduces some new aspects that don't exist in the processing of "attribute-value" data [Zehraoui et al. 2004]. These aspects concern mainly the case representation – the sequences can be very long and with different lengths. Moreover, the huge amount of data and the presence of noise in these data associated to the real time constraints make the case base maintenance of the CBR system necessary.

According to the case representation two different approaches can be distinguished: the cases represented by succession of points and the cases represented by relations between temporal intervals. The representation based on succession of points was proposed in the information research field [Jaczynski and Trousse 1998] while the representation with interval relations was proposed in [Jaere et al. 2002]. The point representation of a case takes into account the temporal information contained in the case in a simple manner. This representation was used successfully in several fields. The representation of case with relations between temporal intervals is a new approach that is not often used. This approach is certainly more complex than the first one, but makes the representation of more precise information in a case possible.

From another point of view cases can be represented as a whole sequence and as a sub-sequence. Several works have used the representation of a case with the whole sequence: a web navigation advisor – Radix system [Corvaisier et al. 1997], the recommender system PADIM [Fuchs et al. 1995] etc. The representation of a case with the sub-sequence was used in the recommendation approach BROADWAY [Jaczynski and Trousse 1998], the CASEP system [Zehraoui et al. 2004] used for sequence prediction and the COBRA system [Malek and Kanawati 2001] that predicts the user actions in a web site. The representation of a case with a whole sequence allows avoiding an additional storage of more precise knowledge. However, in the majority of the systems

that use this representation, no lesson is learned from the different reasoning cycles done during the time series evolution. Moreover, the retrieval of a whole sequence is time consuming. The representation of a case with sub-sequences can require more memory capacity if the whole sequences and the cases are kept in the memory. The advantage of this representation is that, for each reasoning step, useful knowledge is learned by the system.

Another important aspect of applying CBR technology in time series forecasting is the case base maintenance and the control of the size and the content of the system memory. In [Zehraoui et al. 2003], maintenance measurements are associated to the cases in order to take into account the presence of noise in data and to reduce the case base. This leads to the improved CBR system results.

Chapter V

CuBaGe System

A great part of the author's research at the Department of Mathematics and Computer Science (University of Novi Sad) was in the area of implementing decision support systems using by CBR technology. One decision support system called *CaBaGe* (Case-Base Generator) had been already implemented as the part of the author's master thesis [Kurbalija 2006]. The main characteristic of this system is that it is domain independent: the input for the system is 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) by using the solutions of the similar problems from the past. The domain of the cases in the database is not important. Cases could be medical examinations, business data, results of experiments etc, but it is important that all of the cases in one database are from the same domain.

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

The system *CuBaGe* (Curve Base Generator) was a direct result of these intentions [Kurbalija et al. 2009]. The implemented system retrieves curves most similar to a 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 applied successfully in predicting the rhythm of issuing invoices and receiving actual payments at the company "Novi Sad Fair" [Simić et al. 2003], [Simić et al. 2005].

5.1. OVERVIEW OF THE PROBLEM

CuBaGe is a decision support system based on the CBR technology which deals with time series data. Since in many real world domains decisions depend on behaviour of time diagrams, charts or curves, a system that could help experts in their decision making by analysing curves, comparing them to similar curves from the past and predicting the future behaviour of the current curve on the basis of most similar curves from the past could be very useful.

The usual way of time series 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 the time *t*. Depending of the domain a point can be: a result of an experiment, a result of measuring, a value of some goods at a given time etc. When the data is given in this way (as a set of points) it can be represented graphically. Furthermore, if the points are connected they represent a kind of curve.

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 which 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 be computed by using the equation:

$$sim(c,c') = \frac{1}{1+dist(c,c')} \qquad (5.1)$$

This equation satisfies the conditions of the similarity definition and the similarity function given in Chapter 2. Informally, the distance is a dual notion for similarity. It follows from the fact, that computing the similarity directly, is the same as computing the distance first and then (by using this formula) the similarity.

Some authors compute the similarity or distance between curves by a direct composition of distances between points [Faloutsos et al. 1994], [Yi and Faloutsos 2000]. The most frequently used distance measure is the L_2 norm, which is the sum of Euclidian distances between points.

However, the approach in the *CuBaGe* is different: the curve will be approximated by an interpolating polynomial through the set of points. Subsequently, the distance between two curves will be computed 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.

5.2. TIME SERIES REPRESENTATION

The first dilemma in this research was a selection of the appropriate time series representation. A desired property of the system was that time series should be represented as interpolating polynomials between its points. However the problem was to decide: What kind of interpolation should be chosen? If the points are connected just by straight lines then it represents the linear interpolation, but if someone wants smoother curves then some other kind of interpolation with polynomials must be used. The main choice was between the Lagrange interpolating polynomial and the cubic spline because of their simplicity and computational efficiency.

5.2.1. Lagrange Interpolating Polynomial

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 way [Wolfram Mathworld]:

$$P(x) = \sum_{j=1}^{n} P_j(x) \qquad (5.2)$$

where

$$P_j(x) = y_j \prod_{\substack{k=1 \\ k \neq j}}^{n} \frac{x - x_k}{x_j - x_k} \qquad (5.3)$$

When constructing interpolating polynomials, there is a tradeoff between having a better fit and having a smooth well-behaved fitting function. The more data points are used in the interpolation, the higher the degree of the resulting polynomial, and therefore the greater oscillation it will exhibit between the data points. Therefore, a high-degree interpolation may be a poor predictor of the function between points, although the accuracy at the data points will be "perfect."

5.2.2. Cubic Spline

A spline is a piecewise polynomial function that passes through *n+1* control points (x_0,y_0), (x_1,y_1), … ,(x_n,y_n). It can have a very simple form locally, yet it can be globally flexible and smooth at the same time. There are different types of splines. Cubic spline is a spline constructed of piecewise third order polynomials such that the function and its first and second derivates are continuous at the interpolation nodes [Wolfram Mathworld].

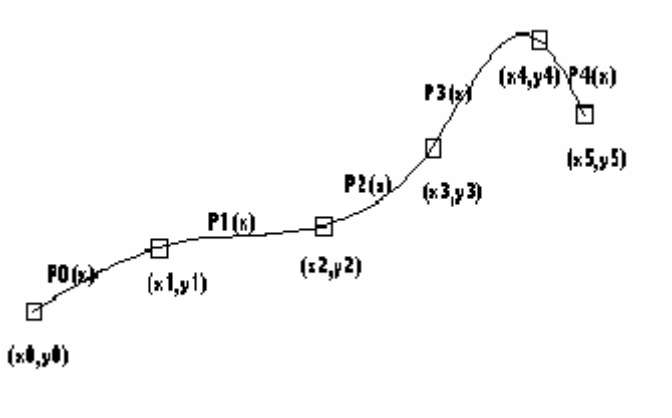

Figure 5.1. Example of a cubic spline

Slika 5.1. Primer kubnog splajna

An example of a cubic spline which passes through 6 points is given in Figure 5.1. This cubic spline consists of 5 different third order polynomials *P0(x), P1(x), …, P4(x)* which "connect" 6 control points.

Generally, a cubic spline which passes through *n+1* control points (x_0,y_0), (x_1,y_1), … ,(x_n,y_n), consists of *n* third order polynomials. If the *i*-th piece of spline is represented in the following way:

$$y_i(x) = a_i + b_i x + c_i x^2 + d_i x^3 \qquad (5.4)$$

then the coefficients can be computed as:

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

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:

$$\begin{bmatrix} 2 & 1 & & & & & \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & 1 & 4 & 1 & & \\ \cdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & 1 & 4 & 1 \\ & & & & & 1 & 2 \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{bmatrix} = \begin{bmatrix} 3(x_1 - x_0) \\ 3(x_2 - x_0) \\ 3(x_3 - x_1) \\ \vdots \\ 3(x_{n-1} - x_{n-3}) \\ 3(x_n - x_{n-2}) \\ 3(x_n - x_{n-1}) \end{bmatrix}. \qquad (5.5)$$

5.2.3. Choice of Time Series Representation

These two kinds of interpolation have the simplest form and 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 the Lagrange interpolating polynomial has the power *n*, while the power of a cubic spline is always less or equal to 3. Polynomials with lower power are simpler and more efficient for calculation and less memory consuming.

- **Oscillation**. The problem of oscillation is shown in Figure 5.2. In pictures a) and b) the Lagrange interpolation polynomial and the cubic spline are shown for the same 6 points, respectively. In the pictures c) and d) the seventh point, which can be considered as an outliner is added (the black square). It can be the result of a bad experiment or a poor measurement. As it can be seen in figure, interpolating polynomial changes significantly (oscillates) while the cubic spline only changes locally. It is considered that the local change is more appropriate for the real world domains.

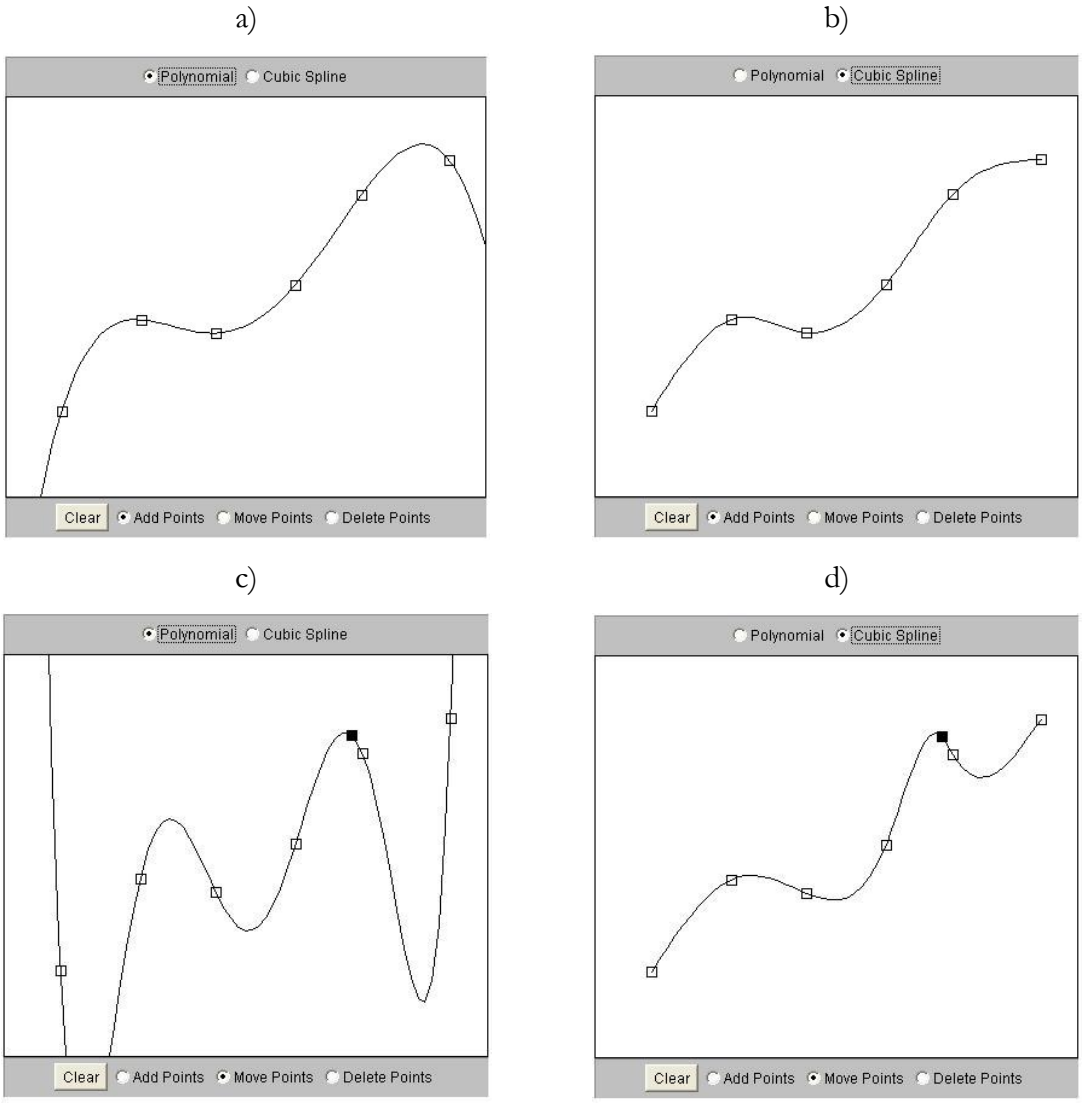

Figure 5.2. Illustration of oscillation

Slika 5.2. Ilustracija oscilacije

5.3. DISTANCE BETWEEN TIME SERIES

As already mentioned, computing the distance (the dual notion for similarity) is sufficient for the case based retrieval. When the curve is given in its analytical form (which is the case with the 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 5.3. This area can be easily calculated by using definite integral. Furthermore, the calculation of definite integral for polynomials is a very simple and efficient operation.

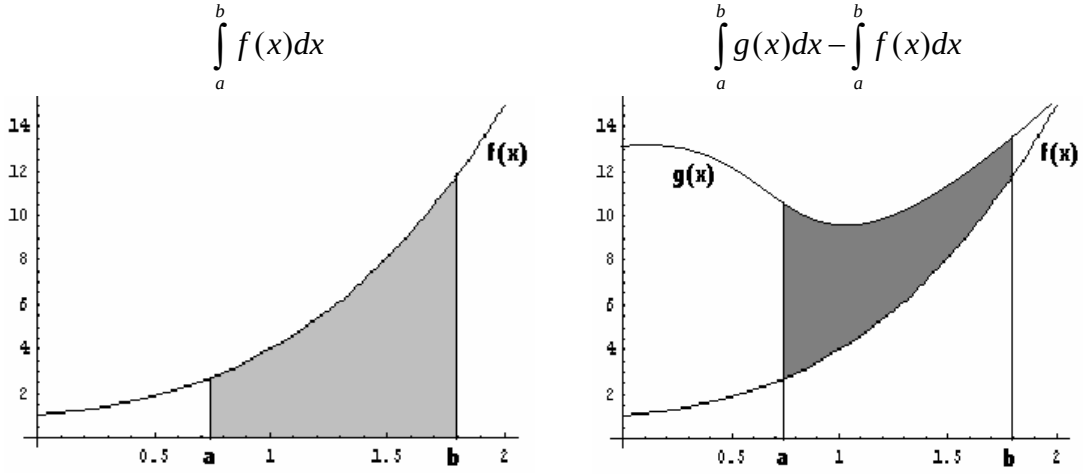

Figure 5.3. Definite integral and surface between curves.

Slika 5.3. Određeni integral i površina između krivih.

For the computation of difference between curves *f* and *g* in the interval *[a,b]* the following equation is used:

$$dist(f,g) = \int_a^b f(x)dx - \int_a^b g(x)dx = \int_a^b (f(x) - g(x))dx = \int_a^b h(x)dx \qquad (5.6)$$

The main difficulty 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 5.4. In the first picture, the curves *f(x)* and *g(x)* are shown; while in the second picture their difference function *h(x)* is shown.

The problem here is that the equation 5.6 gives the result *dist(f,g)=0* and from this follows that *sim(f,g)=1*. This 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 are subtracted instead of added.

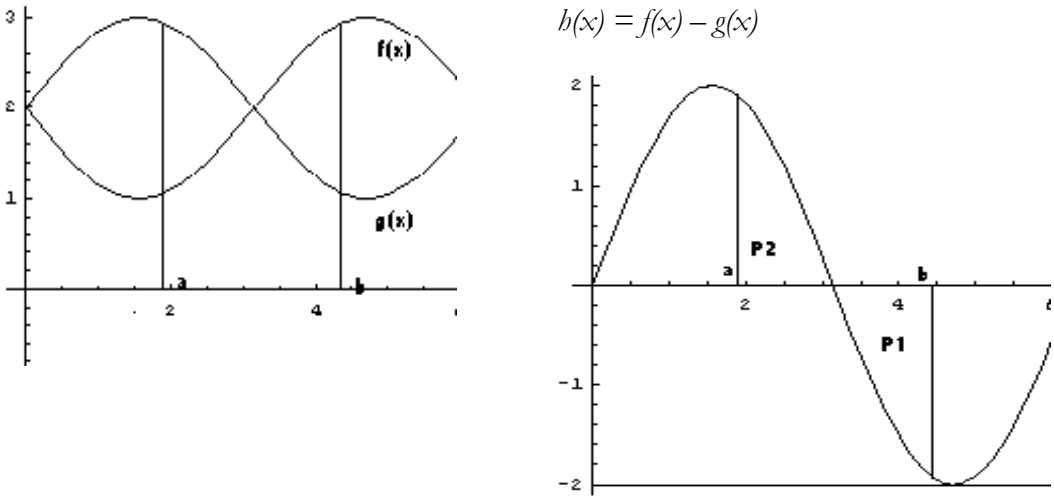

Figure 5.4. Problem with intersection(s).

Slika 5.4. Problem sa presecima.

One solution to 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 the degree 3 is a time consuming operation and requires dealing with complex numbers.

The second solution could be to calculate the square of function *h(x)*. The illustration of this approach can be seen in Figure 5.5. The main idea here is to use the function $h^2(x)$ instead the difference between curves (function *h(x)*). The problem with function *h(x)* is that every time when an intersection occurs, the areas (*P1, P2,…*) 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',…*). The most important 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',…*) using definite integral.

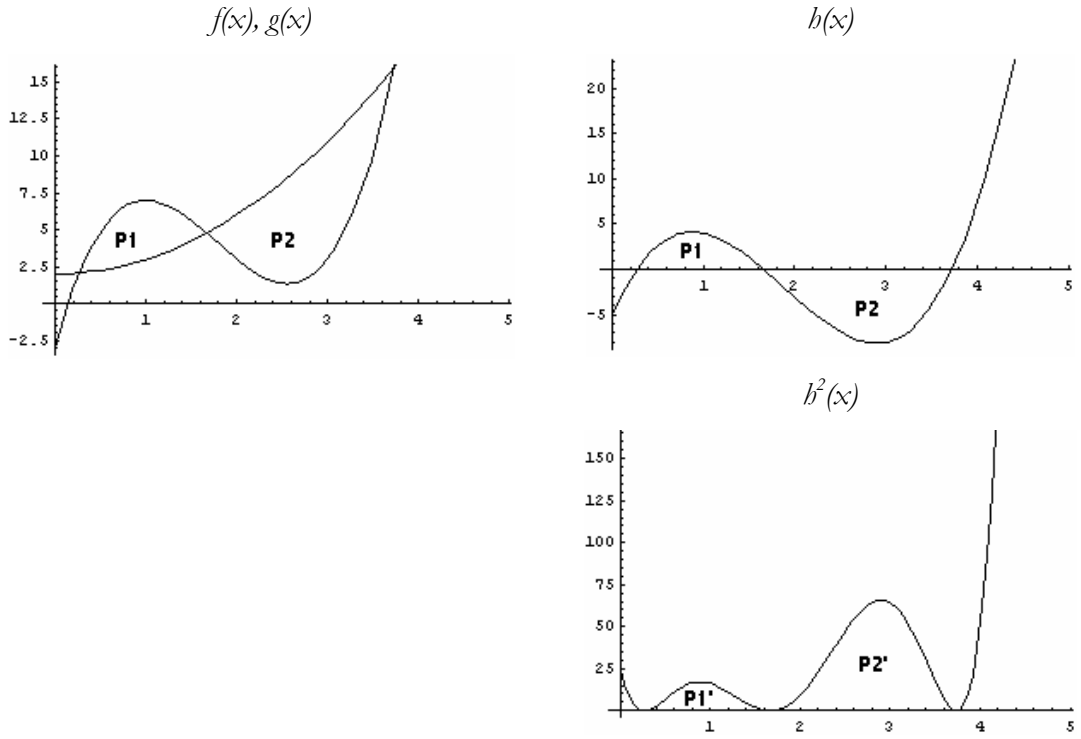

Figure 5.5. Calculation of $h^2(x)$ function.

Slika 5.5. Izračunavanje $h^2(x)$ funkcije.

5.4. IMPLEMENTATION OF *CUBAGE*

The system with the explained characteristics was implemented in Java environment. 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 the cubic spline. The splines are calculated by using the procedure described in part 5.2.2. The distance between two curves is calculated by using the definite integral of the function $h^2(x)$ as described in part 5.3. 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 5.6). 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 using a random walk algorithm. The database consists of 1200 curves. Each curve is given with a set of points. The number of points is also a random number (more than 4 points and less then 10). At the beginning, the system reads those sets of points (from one input file), creates splines for every curve and internally stores the curves. It represents the case base or, rather, the curve base. Then, in the same way, system reads a curve from another input file, and that curve represents a current problem which has to be solved. After that, the system retrieves, from the curve base, the curves which are most similar to the problem curve, and shows the retrieved curves sorted by distance.

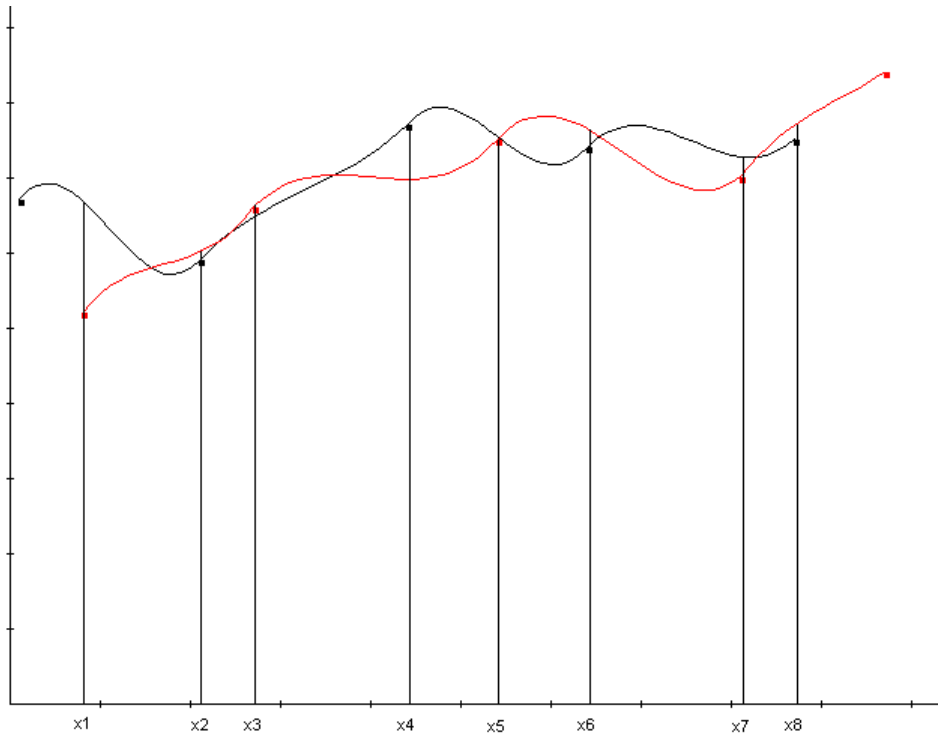

Figure 5.6. Computing the distance between two curves.

Slika 5.6. Izračunavanje rastojanja između dve krive.

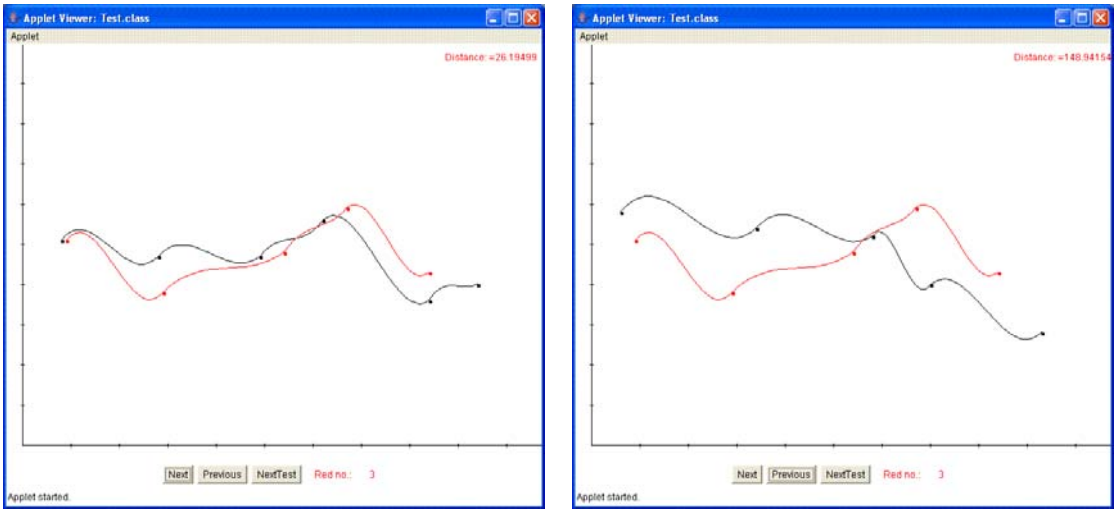

Figure 5.7. Most similar and 33rd similar curve

Slika 5.7. Najsličnija i 33. slična kriva.

Temporarily, a graphical user interface (GUI) was created so that the results of the system can be seen interactively. In Figure 5.7, some snapshots of 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=26,19499). In the right picture the 33rd curve (sorted by distance) is shown (distance=148,94154). As can be seen the system retrieves (intuitively) the most similar curve from the curve base.

CuBaGe was originally designed as an indexing system. For a given problem curve it retrieves the most similar curves from the database and returns them sorted by distance. In this form *CuBaGe* system can be further applied in most of the time series task types: Classification, Clustering, Prediction, Segmentation etc. In the next chapter, the application of the *CuBaGe* system for the prediction in the finance domain will be given.

Chapter VI

CuBaGe in Financial Forecasting

As an application domain for *CuBaGe* system in this thesis, a problem of payment process prediction in the company "Novi Sad Fair" was chosen [Simić et al. 2003], [Simić et al. 2005]. The management of "Novi Sad Fair" wanted to know how high (and in what moment) would be the payment of some services over a future time, with respect to its invoicing.

6.1. The Data

The data for the case base consists of the time series for the payment and invoicing process in the period 2000-2003 for each exhibition. Every year there is between 25 and 30 exhibitions, so 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 in 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 value and stays the same to the end we call *time of saturation* for the invoicing process, and the corresponding value – *value of saturation*.

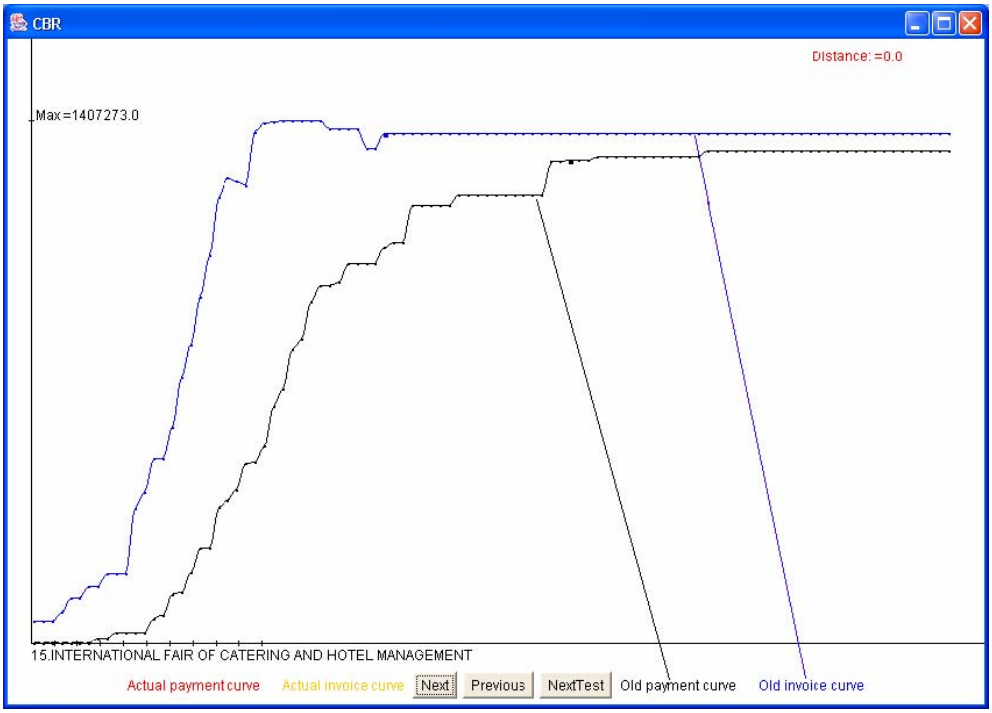

Figure 6.1. Payment and invoice curve.

Slika 6.1. Kriva plaćanja i fakturisanja.

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. Afterwards, 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 – *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 payment value of saturation is between 80% and 100% of invoice value of saturation. One characteristic invoice and a corresponding payment curve are shown in Figure 6.1 as "Old payment curve" and "Old invoice curve". The points of saturation (time and value) are represented by the larger points on the curves.

6.2. CALCULATION OF SOLUTION

The functionality of the *CuBaGe* system looks 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 period 2000-2003 while the other database contains the information about the corresponding payment processes. Each process (invoice or payment) is represented by a set of points; each process is considered as one 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 payment curve at the moment of exhibition. At that moment, the invoice curve is reaching its saturation point, while the payment curve is still far away from its saturation point. These curves are shown in Figure 6.2 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.

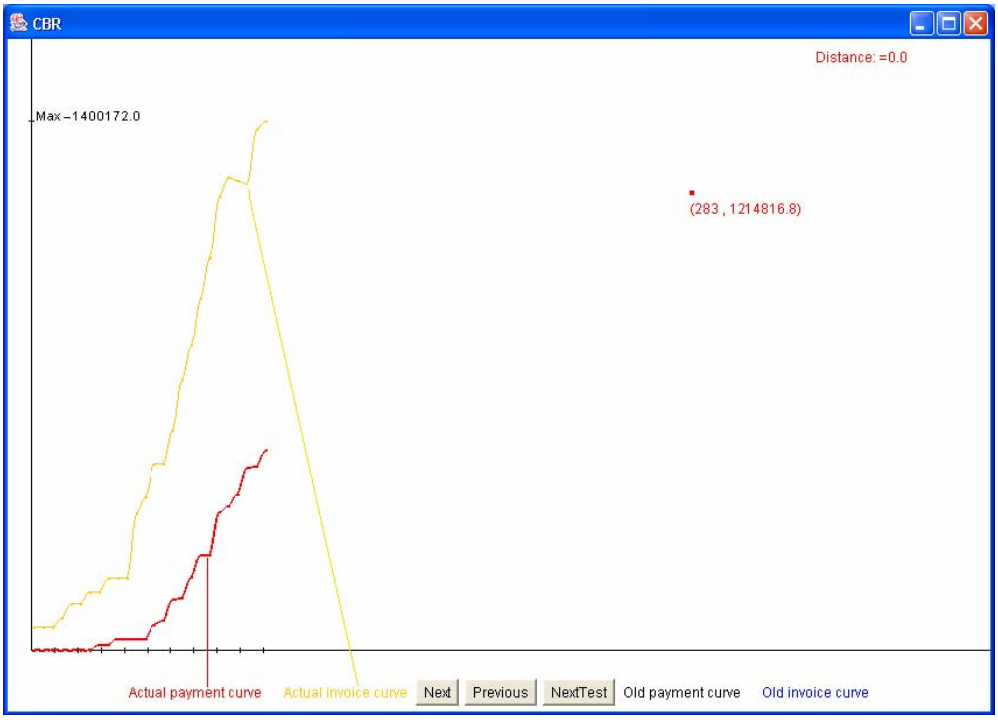

Figure 6.2. Problem payment and invoice curve.

Slika 6.2. Problem kriva plaćanja i fakturisanja.

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 algorithm based on the definite integral described in the previous chapter. Since the values of saturation are different for each exhibition, each curve from the 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:

$$Factor = \frac{actual\_value\_of\_saturation}{old\_value\_of\_saturation} \qquad (6.1)$$

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 value of their "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, the similarity is calculated as shown in equation *(5.1)*.

The goodness for each old payment curve is calculated as:

$$goodness_i = \frac{sim_i}{\sum_{all\_j} sim_j} \qquad (6.2)$$

At the end, the final solution – *payment saturation point*, which is the ordered pair of the time and value of saturation, is calculated as:

$$sat\_point = \sum_{all\_i} goodness_i \cdot sat\_point_i \qquad (6.3)$$

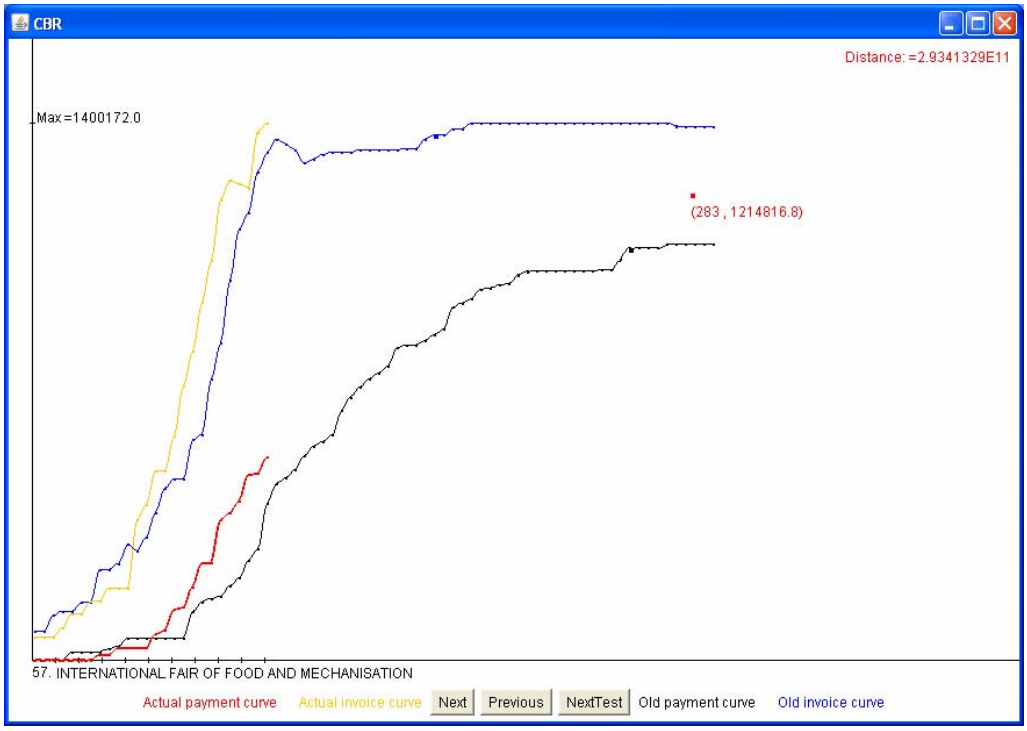

Figure 6.3. Problem payment and invoice curve and some similar curves from database.

Slika 6.3. Problem kriva plaćanja i fakturisanja i neke slične krive iz baze podataka.

The system draws the solution point in the diagram together with the saturation time and value. Solution point can be seen in Figure 6.2 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 may be seen in Figure 6.3.

6.3. EXPERIMENTS

A set of experiments was conducted 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 which have long time intervals without known values) and non-equidistant time series (series in which points are not equally distributed along the time axis).

6.3.1. Standard Time Series

As an illustrative example, the process of calculating the saturation point for the curve number 2 is given in Table 6.1.

Curve nr.	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%**		

Table 6.1. Computing the solution for the curve number 2

Tabela 6.1. Izračunavanje rešenja za krivu broj 2

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 6.1 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 proportional to the similarity value and is calculated using equation 6.2.

The final solution (saturation point of the actual curve) is then calculated as a linear combination of normalized saturation points of these curves. These points together with the real and computed saturation points can be seen in Figure 6.4. The coefficients for linear combination are values of goodness given in equation 6.3. In Table 6.1, the real saturation point and the error made by *CuBaGe* system, is also given.

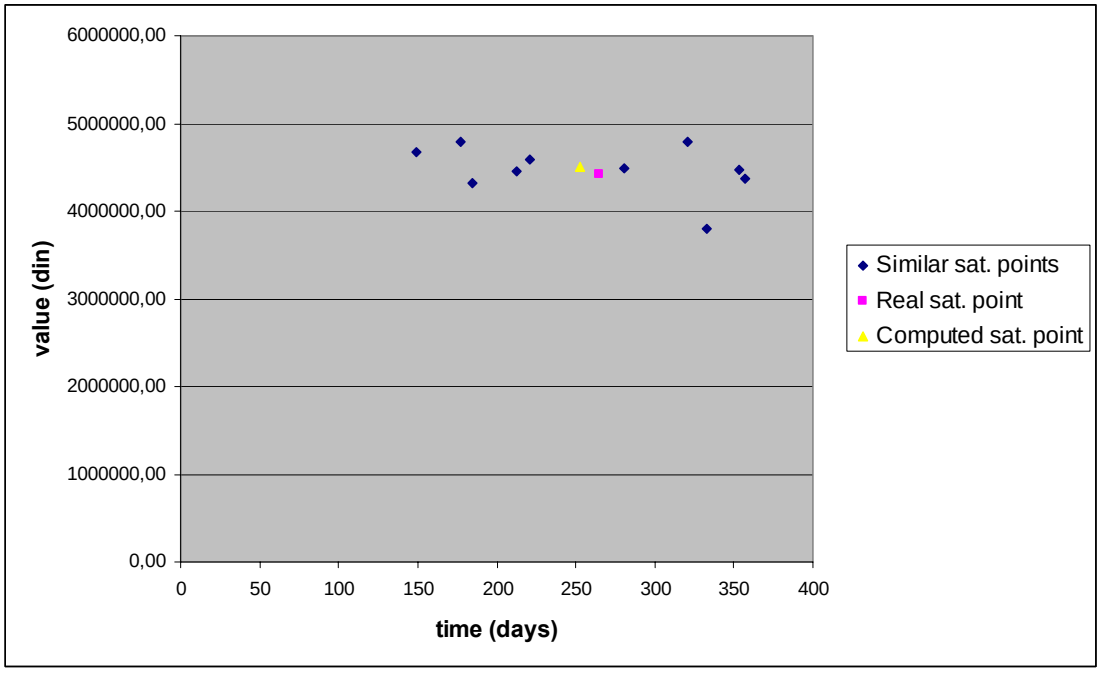

Figure 6.4. Real, computed and similar saturation points from the database.

Slika 6.4. Realna, izračunata i slična tačka zasićenja iz baze podataka.

As the next step, a *K*-fold cross-validation algorithm (where *K*= 10) was conducted in order 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 9 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 which were calculated (made in time dimension and value dimension) are shown in the last row in Table 6.1. All results (errors) from the iterations were then averaged to produce a single estimation.

In Table 6.2 some of the results of the 10-fold cross validation algorithm are given. There, the real saturation points and the saturation points calculated by the *CuBaGe* system are given. In the column "Δ" the variance between real and computed saturation point is shown. After that, the percentage of error for a particular curve is shown. At the end, the average of all errors separately for time and value dimension has been given.

Curve nr.	Real saturation point		Computed saturation point		Δ		Percent of error	
1	289	576998	228.880	602278.750	-60.120	25280.75	15.03%	4.38%
2	265	4425584	252.762	4498063.500	-12.238	72479.50	3.06%	1.64%
3	293	640952	234.418	674823.063	-58.582	33871.06	14.65%	5.28%
4	353	61662912	256.503	61871660.000	-96.497	208748.00	24.12%	0.34%
5	325	11038734	251.434	11018625.000	-73.566	-20109.00	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	440066.000	53.821	85002.00	13.46%	23.94%
9	353	2929711	300.843	2939542.000	-52.157	9831.00	13.04%	0.34%
. . . . .								
. . . . .								
GLOBAL ERROR:							**14.11699%**	**5.87958%**

Table 6.2. Part of the 10-fold cross validation

Tabela 6.2. Deo rezultata „10-fold cross validation" algoritma

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 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 6.1 – "Saturation point"). A similar trend may be observed in the following experiments as well.

6.3.2. Sparse Time Series

In the next phase, 17 experiments were conducted in order to illustrate the ability of the *CuBaGe* system to deal with sparse curves. Here, some points were removed following

this rule: in the i^{th} experiment, only the points with indexes which are the multiples of i are kept. For example, in the 12^{th} experiment only points with these indexes are kept: 0, 12, 24, 36, 48, 60, 72, 84..., while in the 1^{st} 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.

17 is the highest discarding index which gives usable results because with the higher discarding index (more than 17), some of the validation curves, which had to be truncated within the time of the invoice saturation, would have consisted of only one point. This is unusable for any kind of prediction, and furthermore it is impossible to compute spline through a single point.

Experiment	Time error	Value error
F01	14.116990%	5.8795805%
F02	13.040847%	6.7795360%
F03	15.299633%	6.5944070%
F04	14.285990%	6.1562805%
F05	14.445005%	6.3013090%
F06	13.462544%	6.4036360%
F07	15.202986%	6.4354860%
F08	14.777561%	6.8880900%
F09	14.017353%	6.6929040%
F10	14.213959%	6.9920910%
F11	13.713492%	6.7908200%
F12	16.249960%	7.7306930%
F13	14.553808%	7.6972500%
F14	16.167490%	7.0118060%
F15	15.022390%	7.7980690%
F16	16.465824%	7.6579890%
F17	12.756111%	7.0567970%

Table 6.3. Fixed number of removed points

Tabela 6.3. Fiksiran broj izostavljenih tačaka

The results of these experiments are shown in Table 6.3. In the table, the global errors of each 10-fold cross-validation experiment are shown. In Figures 6.5 and 6.6 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.

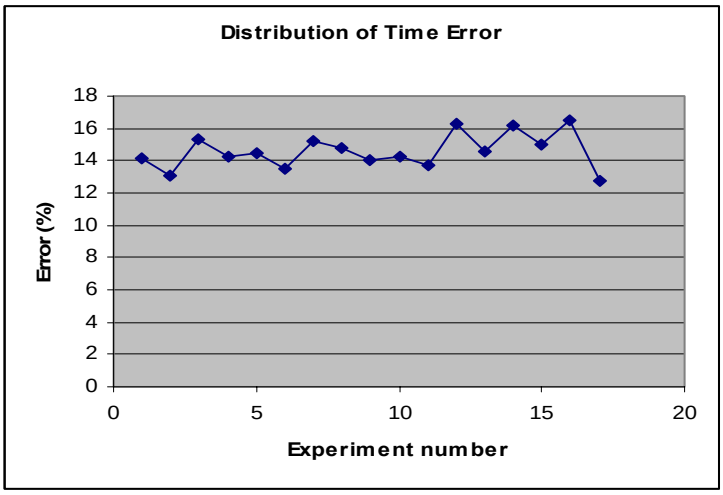

Figure 6.5. Distribution of time error (fixed)

Slika 6.5. Distribucija greške po vremenskoj osi (fiksiran broj izostavljenih tačaka)

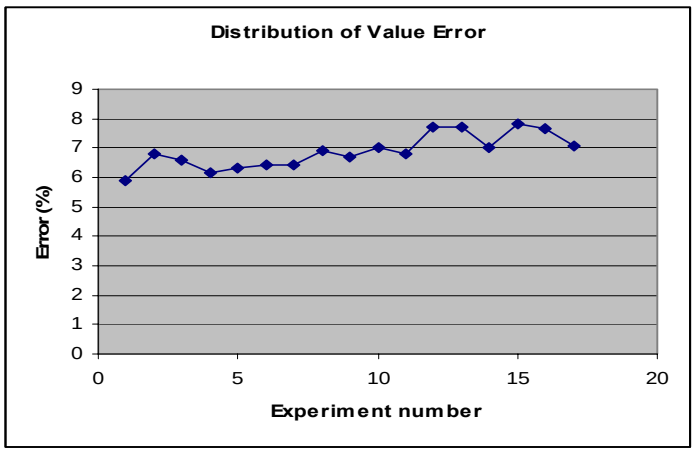

Figure 6.6. Distribution of value error (fixed)

Slika 6.6. Distribucija greške po vrednosnoj osi (fiksiran broj izostavljenih tačaka)

6.3.3. Non-Equidistant Time Series

In the last phase, another 17 experiments were conducted with the intention to illustrate the ability of the *CuBaGe* system to deal with non-equidistant curves. The idea of these experiments is similar to the previous one. In the i^{th} 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 kept points in the 12^{th}

experiment could be: 0, 3, 14, 17, 25, 28, 34, 41, 52, 55... Again, in the 1st experiment none of the points were removed.

Experiment	Time error	Value error
R01	14.116990%	5.8795805%
R02	13.887385%	6.7979000%
R03	16.783285%	7.3578200%
R04	15.951675%	6.9514070%
R05	15.294256%	6.2522780%
R06	15.851670%	6.1262750%
R07	14.489132%	6.6494990%
R08	16.001661%	6.5427990%
R09	13.907600%	8.2518350%
R10	14.316044%	7.2981940%
R11	13.545891%	6.9260440%
R12	15.298765%	6.3968080%
R13	14.793287%	7.7272080%
R14	14.647130%	6.7768580%
R15	16.550610%	6.7215520%
R16	15.571243%	6.3998830%
R17	14.334753%	7.0976140%

Table 6.4. Random number of removed points

Tabela 6.4. Slučajan broj izostavljenih tačaka

In Table 6.4 the results of these experiments are given, while in Figures 6.7 and 6.8 the distributions of time and value errors are given. Again, there is no significant decreasing 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.

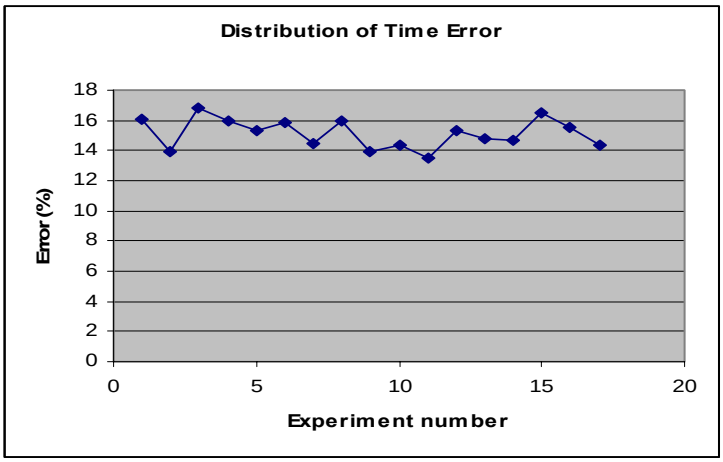

Figure 6.7. Distribution of time error (random)

Slika 6.7. Distribucija greške po vremenskoj osi (slučajan broj izostavljenih tačaka)

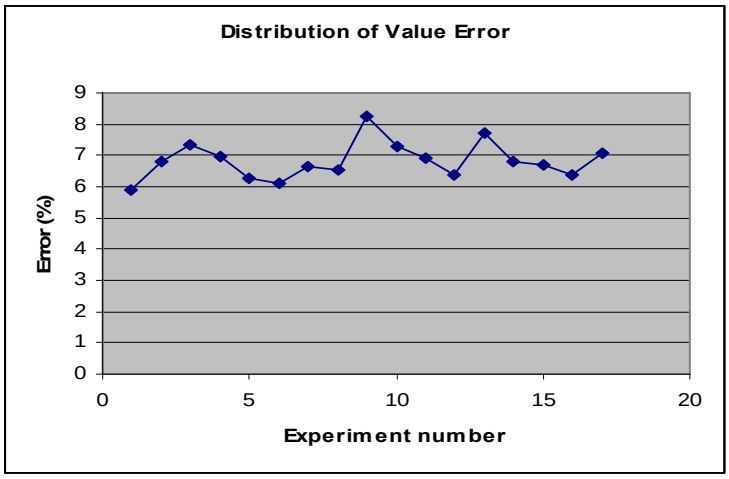

Figure 6.8. Distribution of value error (random)

Slika 6.8. Distribucija greške po vrednosnoj osi (slučajan broj izostavljenih tačaka)

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 100 points, and in experiment F17 each curve consists of only 6 points). All these facts indicate that the system could be applied in other domains, where the data is represented in similar a manner, with a high level of precision and reliability.

Chapter VII

CONCLUSION

Case-based reasoning is a problem solving technique that in many respects is different from the other major AI approaches. Instead of relying solely on a general knowledge of the problem domain, or making associations along generalized relationships between the problem descriptors and conclusions, CBR is able to utilize only specific knowledge of the previously experienced, concrete problem situations (cases). A new problem is solved by finding a similar past case and reusing it in the new problem situation.

The second important difference is that CBR is also an approach to incremental, sustained learning. A new experience is retained each time the problem has been solved, making it immediately available for the future problems. The CBR field has grown rapidly over the last few years, as it may be seen from its increased share of papers at major conferences, or from available commercial tools, successful applications in daily use, and so on.

Case-based reasoning can be used for solving problems in many practical domains such as: mechanical engineering, medicine, business administration, meteorology, physics etc. Furthermore, for each of the domains, various task types can be implemented. Some of

them are: classification, diagnosis, configuration, planning, decision support etc. Also, for each task type all the domains are possible.

Although it is a promising Data Mining technique, CBR was rarely used for time series analysis. Most likely, this is the consequence of the following two reasons: Firstly, the Data Mining techniques are used in time series analysis most recently. Consequently, few scientific efforts were focussed in this direction. Secondly, the use of case-based reasoning for time series processing introduces some new aspects that don't exist in the processing of the "attribute-value" data. These aspects concern mainly the case representation: the sequences may be very long and of different lengths. Moreover, a huge amount of data and the presence of noise in these data associated to the real time constraints make the case base maintenance of the CBR system necessary.

This Thesis tries to "fill the gap" in this research area. A robust and general architecture for curve representation and indexing time series databases, based on Case based reasoning technology, was developed. Also, a corresponding similarity measure was modelled for a given kind of curve representation. The presented architecture may be employed equally well not only in conventional time series (where all values are known), but also in some non-standard time series (sparse, vague, non-equidistant). Dealing with the non-standard time series is the highest advantage of the presented architecture.

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

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

Presented approach utilizes a well known data mining classification technique *k*-nearest neighbour (kNN). In fact, this approach does not calculate the class of the 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 explained as its ability to keep the same accuracy even with the curves whose points are not equidistant. When a random number of points in a random order is removed, the system does 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).

Here 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 numerous advantages. For example, an operational manager can 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 can use this information to plan better possible investments and new exhibitions. This may 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 limited to this case-study only. It can 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, engineering, meteorology etc). The good properties of the system (like 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 will be interesting to test several different interpolation functions.

Also, it will be very interesting to generalise the system so that it can work not only with just a 2-dimensional curves, but also with the 3-dimensional curves and surfaces and in higher dimensions. This may be applicable in domains where decisions depend on more than two factors.

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 to optimize searches in databases. Also, these algorithms (DFT, DWT, SVD…) do not return the exact matches or the most similar time series from the database. Instead, they return only candidates which have to be investigated by using more sophisticated methods. The *CuBaGe* system may be successfully applied in these (postprocessing) situations.

One interesting application of this system may be in a combination with Discrete Fourier Transform or Discrete Wavelet Transform. These approaches are most commonly used in time series indexing. Unfortunately, their disadvantage is that they are defined only for the data where the number of points is an integer power of two. 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 a 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 either for DFT or for DWT can be applied.

Even if the proposed approach with the kNN classification algorithm for the similar time series selection gives high-quality results, it may 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 will 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, medicine, economics), the approach with the Case-Based Reasoning technique is rarely used. This approach, 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 a future research.

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

SAŽETAK

U ovoj doktorskoj disertaciji prikazan je interesantan i perspektivan pristup rešavanja problema analize i predviđanja vremenskih serija korišćenjem Case Based Reasoning (CBR) tehnologije. Detaljno su opisane osnove i glavni koncepti ove tehnologije. Takođe, data je komparativna analiza različitih pristupa u analizi vremenskih serija sa posebnim osvrtom na predviđanje. Kao najveći doprinos ove disertacije, u nastavku je prikazan sistem *CuBaGe* (Curve Base Generator) u kome je realizovan originalni način reprezentacije vremenskih serija zajedno sa, takođe originalnom, odgovarajućom merom sličnosti. Robusnost i generalnost sistema ilustrovana je realnom primenom u domenu finansijskog predviđanja, gde je pokazano da sistem jednako dobro funkcioniše sa standardnim, ali i sa nekim nestandardnim vremenskim serijama (neodređenim, retkim i neekvidistantnim).

Uopšteno govoreći, Case Based Reasoning je tehnika za rešavanje problema, gde se novi problemi rešavaju adaptacijom rešenja koja su odgovarala sličnim problemima u prošlosti. Slučaj se, generalno, može predstaviti kao uređeni par (problem, rešenje), gde prvi elemenat predstavlja opis problema, a drugi uspešno rešenje tog problema iz prošlosti. Osnovni scenario za gotovo sve CBR aplikacije izgleda ovako:

> *Da bi pronašao rešenje aktuelnog problema, sistem traži sličan problem u svojoj "iskustvenoj" bazi, uzima ispravno rešenje najsličnijeg problema iz prošlosti i koristi ga kao osnovu za pronalaženje rešenja aktuelnog problema.*

Glavna prednost ove tehnologije leži u tome što je primenljiva u skoro svakom domenu. CBR sistem ne pokušava da pronađe pravila i uzajamnu povezanost parametara problema. On samo traži slične probleme iz prošlosti i koristi njihova rešenja da bi pronašao rešenje aktuelnog problema. Ovaj pristup je izuzetno pogodan za manje proučene domene – za domene gde veze i pravila između parametara nisu poznati. Druga vrlo važna prednost ove tehnologije leži u tome što je ovakav pristup veoma sličan ljudskim kognitivnim procesima – ljudi svesno ili nesvesno uzimaju u obzir stečeno iskustvo pri rešavanju problema.

Sa druge strane, analiza vremenskih serija je trenutno veoma popularna oblast, što može da se ilustruje velikim brojem objavljenih radova. Ova popularnost može da se objasni velikim brojem domena u kojima se pojavljuju vremenske serije: medicina, industrija, zabava, finansije, računarske nauke, meteorologija i skoro svaka oblast ljudske aktivnosti. Takođe, jedan od pokazatelja velike popularnosti je i čuveni citat [Tufte 1983]:

> *Slučajan uzorak od 4000 slika iz 15 svetskih novina objavljenih između 1974 i 1980 godine pokazuju da je više od 75% slika bilo u stvari grafik neke vremenske serije.*

Tokom istraživanja, izdvojeno je mnoštvo različitih tipova zadataka u oblasti analize vremenskih serija, što je uzrokovano mnogim praktičnim potrebama. Neke od najkarakterističnijih su: indeksiranje, klasifikacija, klasterovanje, predviđanje, sumarizacija, segmentacija itd.

Svi navedeni tipovi zadatka intenzivno koriste meru sličnosti i/ili rastojanja. Indeksiranje i klasterovanje eksplicitno koriste meru sličnosti dok mnogi pristupi klasifikaciji, predviđanju, sumarizaciji itd implicitno koriste meru sličnosti. Neke od najznačajnijih tehnika za izračunavanje mere sličnosti su: Euklidska rastojanja, Dinamičko iskrivljenje vremena, Najduža zajednička sekvenca, Verovatnosni metodi, Generalne transformacije itd.

Baze vremenskih serija su obično veoma velike. Iz toga sledi da je izbor pogodne reprezentacije ili aproksimacije vremenske serije od izuzetne važnosti. Mnogobrojna rešenja koja uglavnom koriste visok nivo apstraktnosti umesto originalnih podatka su razvijena. Neka od najzastupljenijih su: "Discrete Fourier Transform", "Discrete Wavelet Transform", "Singular Value Decomposition", "Piecewise Linear Models", "Piecewise Constant Models", "Adaptive Piecewise Constant Approximation" i "Symbolic Aggregate Approximation".

Svi prethodno navedeni koncepti, tehnike reprezentacije i algoritmi za izračunavanje mere sličnosti su detaljno opisani u disertaciji. Generalno, nemoguće je reći koja od reprezentacija je najbolja. Mnogo realnije je tvrditi da izbor reprezentacije vremenske serije zavisi od domena i od tipa aplikacije. Izbor mere sličnosti zavisi takođe od domena i od tipa aplikacije, ali i od izabrane reprezentacije.

Pored ovih opštih koncepata vezanih za vremenske serije, posebna pažnja u disertaciji je posvećena algoritmima za predviđanje vremenskih serija. Tako su detaljno opisani mnogi metodi predviđanja koji se mogu svrstati u tri velike grupe: Klasični metodi, Uzročni metodi i „Data Mining" metodi.

Kao originalni naučni doprinos u disertaciji je predstavljen sistem *CuBaGe* (Curve Base Generator). Ovaj sistem poseduje neke od mogućnosti analize i predviđanja vremenskih serija, a zasnovan je na CBR tehnologiji. Sistem je u potpunosti implementiran u Java okruženju uz maksimalno korišćenje svih prednosti i koncepata objektno orjentisane tehnologije. U sistemu je realizovan originalni način reprezentacije vremenskih serija zasnovan na splajnovima, zajedno sa, takođe originalnom, odgovarajućom merom sličnosti baziranom na određenom integralu.

Prva dilema tokom modeliranja sistema je bila vezana za izbor pogodne reprezentacije vremenske serije. Želja autora je bila da se vremenske serije predstave u nekom kontinualnom obliku. Kao posledica toga izbor je sužen na neki od interpolacionih polinoma. Interpolacioni polinomi koji imaju najpogodniju formu i najbolje osobine su Lagranžov interpolacioni polinom i splajnovi. Međutim, splajnovi su izabrani zbog sledećih razloga:

- **Stepen polinoma**. Za *n+1* tačku Lagranžov interpolacioni polinom ima stepen *n*, dok je stepen splajna uvek manji ili jednak od *3*. Polinomi sa manjim stepenom su jednostavniji, efikasniji za izračunavanje i zauzimaju manje memorije.
- **Oscilacije.** Kada se u ulaznim podacima pojavi neka tačka koja očigledno ne pripada procesu koji se modelira (što može biti rezultat lošeg eksperimenta ili merenja) tada čitav Lagranžov interpolacioni polinom počinje da osciluje, dok se splajn menja samo lokalno. Ovo je ilustrovano na slici 5.2. Smatra se da je lokalna promena mnogo realnija u stvarnim primenama.

Najveći izazov u CBR tehnologiji ali i u analizi vremenskih serija je odabir pogodne mere sličnosti. To je mera koja će numerički izraziti koliko su dva objekta u posmatranom domenu slična. Alternativno, moguće je računati i rastojanje (dualan koncept sličnosti) dva objekta pa se naknadno korišćenjem formule 5.1 može dobiti sličnost. Velika većina postojećih pristupa izračunava rastojanje dve vremenske serije direktnom kompozicijom rastojanja njihovih tačaka. Najčešće korišćena mera je L_2 norma, koja predstavlja sumu euklidskih rastojanja tačaka. Međutim, u *CuBaGe* sistemu je usvojen drugi princip zasnovan na određenom integralu.

Kad su vremenske serije predstavljene u analitičkoj formi (preko polinoma) tada je moguće koristiti jednu veoma intuitivnu meru rastojanja. Neformalno, rastojanje dve vremenske serije je moguće izračunati kao površinu između njihovih grafika na određenom intervalu. Formalno, ova površina se računa kao određeni integral razlike polinoma koji predstavljaju izabrane vremenske serije.

Kod ovakvog pristupa, javio se problem presečnih tačaka polinoma. Naime, u presečnim tačkama dva polinoma, površine koje predstavljaju rastojanje menjaju znak. Ovo predstavlja problem, jer kada se saberu površine različitih znakova dobija se rezultat različit od očekivanog gde se podrazumeva da su sve površine pozitivne. Ovaj problem u *CuBaGe* sistemu je rešen kvadriranjem razlike polinoma unutar integrala. Ono što je najvažnije, ovakvim rešenjem nije narušen poredak veličina površina, a sve površine „postaju pozitivne“.

Za domen aplikacije *CuBaGe* sistema odabran je domen finansijskog predviđanja u kompaniji „Novosadski sajam“. Za svaku izložbu sajma praćeni su procesi fakturisanja i plaćanja na svaka 4 dana. Kako ovo praćenje traje oko 400 dana vremenske serije imaju

oko 100 tačaka. Svaka tačka sadrži vreme merenja i vrednost fakturisanja/plaćanja. Analizom ovih podataka utvrđeno je da i kriva fakturisanja i kriva plaćanja u jednom trenutku dostignu tačku „zasićenja“ (tačka posle koje se vrednosti ne menjaju više od 10%). Za krivu fakturisanja ova tačka se nalazi u trenutku izložbe, dok se za krivu plaćanja nalazi znatno kasnije. Takođe, krajnja vrednost plaćanja nikad ne dostigne krajnju vrednost fakturisanja, uvek je između 80% i 100%.

U ovakvom domenu, problem je definisan kao predviđanje tačke saturacije procesa plaćanja u trenutku saturacije fakturisanja (u trenutku izložbe). Sve vremenske serije (i fakturisanja i plaćanja) u sistemu su predstavljene preko splajnova. Za unet problem (kriva fakturisanja i plaćanja u trenutku izložbe) sistem prvo izdvaja 10% najsličnijih kompletnih krivih iz baze podatka. Sličnost se računa preko rastojanja, a rastojanja se izračunavaju po već opisanom postupku preko određenih integrala. Nakon ovoga, sistem izračunava tačku zasićenja procesa plaćanja preko težinske sume tačaka zasićenja pronađenih 10% najsličnijih vremenskih serija.

U disertaciji je sproveden i niz eksperimenata kako bi se utvrilo ponašanje sistema u različitim situacijama. Kao standardni Data Mining postupak, sprovedena je tehnika *K-fold cross-validation*, gde je *K* jednako 10. Originalna baza podataka je podeljena u 10 podskupova jednake veličine. Jedan od 10 podskupova je izdvojen kao skup za proveru, dok su ostalih 9 korišćeni kao podaci za obuku sistema. Ovaj algoritam se ponavlja 10 puta, gde se svaki od podskupova uzima tačno jednom kao skup za proveru. U svakoj iteraciji algoritma meri se relativna greška stvarne vrednosti u odnosu na izračunatu po vremenskoj i po vrednosnoj osi za svaku vremensku seriju iz skupa za proveru. Rezultati su prikazani u tabeli 6.2 i dobijeno je da je greška po vremenskoj osi 14.11699% dok je greška po vrednosnoj osi 5.87958%. Veća greška po vremenskoj osi je potpuno razumljiva jer vreme saturacije mnogo više zavisi od mnogih ekonomskih i političkih faktora. Ovakav trend će se javljati i u narednim eksperimentima.

U sledećoj fazi izvedeno je 17 eksperimenata koji pokazuju sposobnost sistema za rad sa retkim vremenskim serijama. Ovde su neke tačke uklonjene iz vremenskih serija prema sledećem pravilu: u *i*-tom eksperimentu zadržane su samo tačke čiji su indeksi umnošci od *i*. Sa ovakvim vremenskim serijama je opet izvršen *K-fold cross-validation* algoritam.

Rezultati su vidljivi u tabeli 6.3. Greške po vremenskoj osi se kreću od 12.756111% do 16.249960%, dok se greške po vrednosnoj osi kreću od 5.8795805% do 7.7980690%. Ono što je još bitnije, nije primećen neki značajniji rast grešaka kako je odstranjivan sve veći broj tačaka (slike 6.5 i 6.6).

U sledećoj fazi izvedeno je još 17 eksperimenata koji pokazuju sposobnost sistema za rad sa ne-ekvidistantnim vremenskim serijama. Ovde su takođe neke tačke uklanjane ali po sledećem pravilu: u *i*-tom eksperimentu prva tačka je zadržana, a nakon nje slučajan broj tačaka (iz intervala *[0, i]*) je uklonjen; sledeća tačka je zadržana a onda opet slučajan broj tačaka uklonjen itd. Na primer indeksi zdržanih tačaka u 12-tom eksperimentu mogli bi biti: 0, 3, 14, 17, 25, 28, 34, 41, 52, 55... Rezultati su vidljivi u tabeli 6.4. Greške po vremenskoj osi se kreću od 13.545891% do 16.783285%, dok se greške po vrednosnoj osi kreću od 5.8795805% do 8.2518350%. I u ovoj fazi rezultati su obećavajući: nije primećen neki značajniji rast grešaka kako je odstranjivan sve veći broj tačaka u slučajnom poretku (slike 6.7 i 6.8).

Navedeni eksperimenti demonstriraju generalnost i stabilnost *CuBaGe* sistema u ovom domenu. Sistem je imao iste rezultate sa standardnim vremenskim serijama (gde su tačke ekvidistantne i relativno „guste“) kao i sa retkim i neekvidistantnim serijama gde je velika količina tačaka odstranjena (originalne vremenske serije se sastoje od 100 tačaka dok se u eksperimentu F17 sastoje od samo 6). Sve ove činjenice indikuju da se sistem može upotrebiti i u drugim domenima sa visokim nivoom stabilnosti i preciznosti.

Ova disertacija pokušava da popuni prazninu u istraživanju vremenskih serija posebno putem CBR tehnologije. Razvijena je robusna arhitektura za reprezentaciju i indeksiranje vremenskih serija bazirana na CBR tehnologiji. Takođe, za predloženu reprezentaciju vremenskih serija, odgovarajuća mera sličnosti/rastojanja je modelovana. Opisana arhitektura može biti jednako dobro primenjena i sa retkim i neekvidistantnim vremenskim serijama kao i sa standardnim.

Predloženi matematički koncept za reprezentaciju vremenskih serija – splajnovi – je veoma konzistentan i fleksibilan. Splajnovi takođe imaju i niz drugih prednosti: jednostavnost konstrukcije, preciznost izračunavanja, mogućnost aproksimacije kompleksnih oblika itd.

Osim toga, predložena mera sličnosti/rastojanja koja koristi određene integrale je najpodobnija za reprezentaciju putem splajnova i garantuje visok nivo preciznosti i ispravnosti. Izračunavanje određenih integrala je veoma efikasno, a istovremeno ova mera bazirana na površinama je jednostavna i intuitivna.

Predstavljeni sistem nije ograničen samo na ovaj domen primene. *CuBaGe* sistem može biti primenjen i na druge ekonomske parametre kao na primer: izdaci, investicije, profit itd. Prored toga, sistem može biti upotrebljen i u drugim ne-ekonomskim domenima gde neke odluke zavise od ponašanja vremenskih serija (medicina, fizika, inžinjerstvo, meteorologija itd.). Dobre osobine sistema kao što su generalnost, stabilnost, preciznost i mogućnost za rad sa neekvidistantnim i retkim vremenskim serijama otvaraju širok spektar mogućih domena.

REFERENCES

[Aach and Church 2001] Aach, J. and Church, G. *Aligning gene expression time series with time warping algorithms*. Bioinformatics; 2001, Volume 17, pp. 495-508.

[Aamodt, Plaza 1994] Aamodt, A., Plaza, E., (1994), "Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches", *AI Commutations*, pp. 39-58.

[Acorn and Walden 1992] Acorn, T., Walden, S. *SMART Support management automated reasoning technology for Compaq customer service*. Tenth National Conference on AI. MIT Press. (1992)

[Aggarwal et al. 2001] Aggarwal, C., Hinneburg, A., Keim, D. A. *On the surprising behavior of distance metrics in high dimensional space*. In proceedings of the 8th International Conference on Database Theory; 2001 Jan 4-6; London, UK, pp 420-434.

[Agrawal et al. 1993] Agrawal, R., Faloutsos, C., Swami, A. *Efficient Similarity Search in Sequence Data bases*. International Conference on Foundations of Data Organization (FODO); 1993.

[Agrawal et al. 1995] Agrawal, R., Lin, K.-I., Sawhney, H.S., Shim, K. *Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Trime-Series Databases*. Proceedings of 21St International Conference on Very Large Databases; 1995 Sep; Zurich, Switzerland, pp. 490-500.

[Arsham-Web] Hossein Arsham, *Time-Critical Decision Making for Business Administration*, Internet source: http://home.ubalt.edu/ntsbarsh/stat-data/Forecast.htm

[Bacha and Meyer 1992] Bacha, H., Meyer, W., (1992) *A neural network architecture for load forecasting*. In: Proceedings of the IEEE International Joint Conference on Neural Networks, 2, pp. 442–447.

[Balestrino et al. 1994] Balestrino, A., Bini Verona, F., Santanche, M., (1994) Time series analysis by neural networks: Environmental temperature forecasting. Automazione e Strumentazione 42 (12), pp.81–87.

[Bartsch-Spörl 1999] Bartsch-Spörl, B., Lenz, M., and Hübner, A. 1999. Case-Based Reasoning: Survey and Future Directions. In *Proceedings of the 5th Biannual German Conference on Knowledge-Based Systems: Knowledge-Based Systems - Survey and Future Directions* (March 03 - 05, 1999). F. Puppe, Ed. Lecture Notes In Computer Science, vol. 1570. Springer-Verlag, London, 67-89.

[Berndt and Clifford 1996] Berndt, D.J., Clifford, J. *Finding Patterns in Time Series: A Dynamic Programming Approach*. In Advances in Knowledge Discovery and Data Mining AAAIIMIT Press, Menlo Park, CA, 1996, pp. 229-248.

[Bianchi et al. 1999] Bianchi M., Boyle M., Hollingsworth D., *A comparison of methods for trend estimation*, Applied Economics Letters (1999), 6(2): 103-109.

[Bollobas et al. 2001] Bollobas, B., Das, G., Gunopulos, D., Mannila, H. *Time-Series Similarity Problems and Well-Separated Geometric Sets*. Nordic Jour. of Computing 2001; 4.

[Boser et al. 1992] B. E. Boser, I. M. Guyon, and V. N. Vapnik. *A training algorithm for optimal margin classifiers*. In D. Haussler, editor, 5th Annual ACM Workshop on COLT, pages 144-152, Pittsburgh, PA, 1992. ACM Press.

[Box and Jenkins 1970] George Box and Gwilym Jenkins (1970) *Time series analysis: Forecasting and control*, San Francisco: Holden-Day.

[Bradburn and Zeleznikow 1993] Bradburn C., Zeleznikow J., 1993, *The Application of Case Based Reasoning to the Task of Health Care Planning,* Proceedings of the First European Workshop on Case Based Reasoning (EWCBR-93), Lecture Notes In Artificial Intelligence 837, Pages 365-378.

[Brezinski and Redivo Zaglia 1991] C. Brezinski and M. Redivo Zaglia, *Extrapolation Methods. Theory and Practice*, North-Holland, 1991.

[Burkhard 2001] Burkhard, H. 2001. Similarity and distance in case based reasoning. *Fundam. Inf.* 47, 3-4 (Aug. 2001), 201-215.

[Burkhard, Richter 2001] Burkhard, H. and Richter, M. M. 2001. On the notion of similarity in case based reasoning and fuzzy theory. In *Soft Computing in Case Based Reasoning*, S. K. Pal, T. S. Dillon, and D. S. Yeung, Eds. Springer-Verlag, London, 29-45.

[Cao and Tay 2001] Cao LJ, Tay FEH. *Financial forecasting using support vector machines*. Neural Computing Applications 2001; 10. pp. 184–192.

[Chakrabarti et al. 2002] Chakrabarti, K., Keogh, E., Pazzani, M., Mehrotra, S. *Locally adaptive dimensionality reduction for indexing large time series databases*. ACM Transactions on Database Systems. Volume 27, Issue 2, (June 2002). pp 188-228.

[Chan and Fu 1999] Chan, K., Fu, A.W. *Efficient time series matching by wavelets.* Proceedings of 15th IEEE International Conference on Data Engineering; 1999 Mar 23-26; Sydney, Australia, pp. 126-133.

[Cheetham 2005] Cheetham, W., *Tenth Anniversary of Plastics Color Matching*, Artificial Intelligence Magazine, Volume 26, No. 3, (2005). pp 51-61.

[Cheetham and Goebel 2007] Cheetham, W., Goebel, K.: *Appliance Call Center: A Successful Mixed-Initiative Case Study*, Artificial Intelligence Magazine, Volume 28, No. 2, (2007) 89 – 100.

[Corvaisier et al. 1997] S. F. Corvaisier, A. Mille, and J. M. Pinon. *Information retrieval on the world wide web using a decision making system.* Actes de la 5eme conférence sur la Recherche d'Informations Assistée par Ordinateur sur Internet (RIAO'97), Centre des hautes études internationales d'Informatique, Montréal, pages 285–295, 1997.

[De Groot and Wurtz 1991] De Groot, C., Wurtz, D., (1991) *Analysis of univariate time series with connectionist nets: a case study of two classical examples.* Neurocomputing 3, 177–192.

[Debregeas and Hebrail 1998] Debregeas, A., Hebrail, G. *Interactive interpretation of kohonen maps applied to curves.* In proceedings of the 4th Int'l Conference of Knowledge Discovery and Data Mining; 1998 Aug 27-31; New York, NY, pp 179-183.

[Faloutsos et al. 1994] Faloutsos, C., Ranganathan, M., Manolopoulos, Y. *Fast subsequence matching in time-series databases.* In proceedings of the ACM SIGMOD Int'l Conference on Management of Data; 1994 May 25-27; Minneapolis, MN, pp 419-429.

[Fuchs et al. 1995] B. Fuchs, A. Mille, and B. Chiron. *Operator decision aiding by adaptation of supervision strategies.* Case-Based Reasoning Research and Development, proceedings of the 1st International Conference on Case-Based Reasoning (ICCBR'95), LNAI, 1010:23–32, 1995.

[Gately 1996] Gately, E., (1996) *Neural Networks for Financial Forecasting.* John Wiley, New York.

[Ge and Smyth 2000] Ge, X., Smyth, P. *Deformable Markov Model Templates for Time-Series Pattern Matching.* Proceedings of 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2000 Aug 20-23; Boston , MA, pp. 81-90.

[Geurts 2001] Geurts, P. *Pattern extraction for time series classification.* Proceedings of Principles of Data Mining and Knowledge Discovery, 5th European Conference; 2001 Sep 3-5; Freiburg, Germany, pp 115-127.

[Gierl et al. 2003] Gierl L, Steffen D, Ihracky D, Schmidt R, 2003, *Methods, architecture, evaluation and usability of a case-based antibiotics advisor.* Computer Methods and Programs in Biomedicine 72 (2003) 139-154

[Gierl, Stengel-Rutkowski 1992] Gierl L., Stengel-Rutkowski S.,1992 *Prototypes as a Core Representation of Medical Knowledge In Diagnosis and Knowledge Acquisition.* In Lun et al, MEDINFO 92, Pages 1088-1094.

[Gierl, Stengel-Rutkowski 1994] Gierl L., Stengel-Rutkowski S., 1994, *Integrating Consultation and Semi-automatic Knowledge Acquisition in a Prototype-based Architecture:*

Experiences with Dysmorphic Syndromes, Artificial Intelligence in Medicine 6, Pages 29-49.

[Guralnik and Srivastava 1999] Guralnik, V., Srivastava, J. *Event detection from time series data.* In proceedings of the 5th ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining; 1999 Aug 15-18; San Diego, CA, pp 33-42.

[Guttman 1984] Guttman, A. *R-trees: A dynamic index structure for spatial searching.* In Proc. ACM SIGMOD Conf. 1984, pp 47-57.

[Hamilton 1994] Hamilton J.D. *Time Series Analysis*, Princeton University Press, 1994, ISBN 0691042896.

[Han and Kamber 2006] Jiawei Han, Micheline Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006, ISBN 1558609016, 9781558609013.

[Hastie et al. 2001] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Series in Statistics. Springer-Verlag, 2001.

[Heider 1996] Heider, R. 1996. *Troubleshooting CFM 56-3 Engines for the Boeing 737 - Using CBR and Data-Mining.* In Proceedings of the Third European Workshop on Advances in Case-Based Reasoning (November 14 - 16, 1996). I. Smith and B. Faltings, Eds. Lecture Notes In Computer Science, vol. 1168. Springer-Verlag, London, 512-518.

[Hinkle and Toomey 1994] Hinkle, D., and Toomey, C. N., *CLAVIER: Applying case-based reasoning on to composite part fabrication.* Proceeding of the Sixth Innovative Application of AI Conference, Seattle, WA, AAAI Press, (1994). pp. 55-62.

[Hochheiser and Shneiderman 2001] Hochheiser, H., Shneiderman,, B. *Interactive Exploration of Time-Series Data.* Proceedings of 4th International conference on Discovery Science; 2001 Nov 25-28; Washington, DC, pp. 441-446.

[Hornik et al. 1989] Hornik, K., Stinchcombe, M., White, H., (1989) *Multilayer feedforward networks are universal approximators.* Neural Networks 2, pp. 359–366.

[Huang et al. 2005] Huang, W., Nakamori, Y., and Wang, S. 2005. *Forecasting stock market movement direction with support vector machine.* Comput. Oper. Res. 32, 10 (Oct. 2005), pp. 2513-2522.

[Iglezakis 2001] Iglezakis, I. (2001), "The Conflict Graph for Maintaining Case-Based Reasoning Systems", *4th International Conference on Case-Based Reasoning (ICCBR 2001),* pp. 263-276, Vancouver, Canada, July/August 2001.

[Indyk et al. 2000] Indyk, P., Koudas, N., Muthukrishnan, S. *Identifying representative trends in massive time series data sets using sketches.* In proceedings of the 26th Int'l Conference on Very Large Data Bases; 2000 Sept 10-14; Cairo, Egypt, pp 363-372.

[Ivanović et al. 2002] 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, September 11-13, 2002, pp. 255-264.

[Jaczynski and Trousse 1998] M. Jaczynski and B. Trousse. *www assisted browsing by reusing past navigations of a group of users.* Proceedings of EWCBR'98, LNAI, 1488:160–171, 1998.

[Jaere et al. 2002] M D. Jaere, A. Aamodt, and P. Skalle. *Representing temporal knowledge for case-based prediction*. ECCBR'02, 2002.

[Kahn and Anderson 1994] Kahn C. E. J., Anderson G. M., 1994, *Case Based Reasoning and Imaging procedure Selection,* Investigative Radiology 29, Pages 643-647.

[Kahveci and Singh 2001] Kahveci, T., Singh, A. *Variable length queries for time series data.* In proceedings of the 17th Int'l Conference on Data Engineering; 2001 Apr 2-6; Heidelberg, Germany, pp 273-282.

[Kalpakis et al. 2001] 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; 2001 Nov 29-Dec 2; San Jose, CA, pp 273-280.

[Kamp, Pirk, Burkhard 1996] Kamp, G., Pirk, P., and Burkhard, H. 1996. FALLDATEN: Case-Based Reasoning for the Diagnosis of Technical Devices. In *Proceedings of the 20th Annual German Conference on Artificial intelligence: Advances in Artificial intelligence* (September 17 - 19, 1996). G. Görz and S. Hölldobler, Eds. Lecture Notes In Computer Science, vol. 1137. Springer-Verlag, London, 149-161.

[Keogh 2002] Keogh, E. *Exact indexing of dynamic time warping.* Proceedings of 2gth Internation Conference on Very Large Databases; 2002; Hong Kong, pp. 406-417.

[Keogh and Pazzani 1998] 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; 1998 Aug 27-31; New York, NY, pp 239-241.

[Keogh et al. 2001] Keogh, E., Chakrabarti, K., Mehrotra, S., Pazzani, M. *Locally adaptive dimensionality reduction for indexing large time series databases.* Proceedings of ACM SIGMOD International Conference; 2001.

[Keogh et al. 2002] Keogh, E., Lonardi, S., Chiu, W. *Finding Surprising Patterns in a Time Series Database In Linear Time and Space.* In the gth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2002 Jul 23 - 26; Edmonton, Alberta, Canada, pp 550-556.

[Kettler et al. 1994] Kettler, B.P.; Hendler, J.A.; Andersen, W.A.; Evett, M.P. *Massively parallel support for case-based planning*, IEEE Expert, Volume 9, Issue 1, Feb 1994 Page(s): 8 – 14

[Korn et al. 1997] Kom, F., Jagadish, H., Faloutsos, C. *Efficiently supporting ad hoc queries in large datasets of time sequences.* Proceedings of SIGMOD International Conferences 1997; Tucson, AZ, pp. 289-300.

[Koton 1988] Koton P., 1988 *Reasoning about Evidence in Casual Explanation*, Proceedings of the Case Based Reasoning Workshop (1988), Morgan Kaufman Publishers, Pages 260-207.

[Kovacic et al. 1992] Kovacic K., Sterling L., Petot G., Ernst G., Yang N., 1992, *Towards an Intelligent Nutrition Manager,* In Proceedings ACM/SIGAPP Symposium on Computer Applications, Pages 1293-1296.

[Kurbalija 2003] Kurbalija V., Ivanović M., *Artificial Intelligent Support for Medical Diagnosis,* Proceedings of a Workshop On Computational Intelligence and

Information Technologies, Niš, Serbia and Montenegro, October 13, 2003, pp. 69-73.

[Kurbalija 2006] Kurbalija V., *Implementation of Decision Support Systems using Case Based Reasoning Technology*, Master Thesis, Novi Sad, 2006.

[Kurbalija et al. 2009] Kurbalija V., Ivanović M., Budimac Z., *CASE-BASED CURVE BEHAVIOUR PREDICTION*, Software: Practice and Experience, ISSN: 0038-0644, Volume 39, Issue 1, January 2009, pp 81-103 (DOI: 10.1002/spe.891).

[Kurbalija, Ivanović 2003] Kurbalija V., Ivanović M., *CBG – A Framework for Intelligent Decision Support Systems*, Proceedings of Fourth International Conference on Informatics and Information Technology "Molika 2003", Bitola, Macedonia, December 11-14, 2003, pp. 118-128.

[Lenz 1998] Lenz, M. 1998. Defining Knowledge Layers for Textual Case-Based Reasoning. In *Proceedings of the 4th European Workshop on Advances in Case-Based Reasoning* B. Smyth and P. Cunningham, Eds. Lecture Notes In Computer Science, vol. 1488. Springer-Verlag, London, 298-309.

[Lenz et al. 1998] Lenz, M., Brtsch-Sporl, B., Burkhard, H., Wess, S. (1998), *Case-Based Reasoning Technology: From Foundation to Applications*, Springer, 1998.

[Lenz, Burkhard 1996] Lenz, M. and Burkhard, H. 1996. Case Retrieval Nets: Basic Ideas and Extensions. In *Proceedings of the 20th Annual German Conference on Artificial intelligence: Advances in Artificial intelligence* (September 17 - 19, 1996). G. Görz and S. Hölldobler, Eds. Lecture Notes In Computer Science, vol. 1137. Springer-Verlag, London, 227-239.

[Lenz, Hübner 1998] Mario Lenz, Andre Hübner, Mirjam Kunze: *Question Answering with Textual CBR* Proc. International Conference on Flexible Query Answering Systems, May 1998, Roskilde, Denmark.

[Li et al. 1990] Li, M., Mehrotra, K., Mohan, C., Ranka, S., (1990) *Sunspots numbers forecasting using neural networks.* In: Proceedings of 5th IEEE International Symposium on Intelligent Control, pp. 524–529.

[Lin et al. 2003] 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, 8th ACM SIGMOD; 2003 Jun 13; San Diego, CA.

[Lin et al. 2004] Lin, J., Keogh, E., Lonardi, S., Lankford, J. P., Nystrom, D. M. *Visually Mining and Monitoring Massive Time Series.* Proceedings of the loth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining; 2004 Aug 22-25; Seattle, WA.

[Linear prediction Web] Linear Prediction at http://www.phon.ucl.ac.uk/courses/spsci/dsp/lpc.html

[Makridakis et al. 1998] Makridakis, S., S.C. Wheelwright, and R.J. Hyndman (1998) *Forecasting: methods and applications*, New York: John Wiley & Sons.

[Malek and Kanawati 2001] M. Malek and R. Kanawati. Cobra: *A cbr-based approach for predicting users actions in web site.* 3rd International Conference on Case-based Reasoning ICCBR'01., pages 336–346, 2001.

[Minor 1998] Mirjam Minor: *Managing Test Specifications with Case-Based Reasoning* in: H.-D. Burkhard, L. Czaja, P. Starke (Eds.): Proceedings of the Workshop on Concurrency, Specification and Programming (CS&P-98), Humboldt University, Berlin, 1998.

[Minor 1999] Mirjam Minor: *Managing Test Specifications with Case-Based Reasoning.* in: E. Melis (Ed.): 7th German Workshop on CBR, Universität Saarbrücken, 1999.

[Minor 2000] Mirjam Minor, Alexandre Hanft: *The Life Cycle of Test Cases in a CBR System.* Proc. EWCBR-2000, LNAI 1898, 455-466, Springer Verlag, 2000.

[Minor, Carlos 2000] Mirjam Minor, Jose Carlos del Prado: *Multilingual Textual Case-Based Reasoning.* In: H.-D. Burkhard, L. Czaja, A. Skowron, P. Starke (Eds.): Proc. Workshop CS&P, Informatik-Bericht Nr. 140 der HU Berlin, Vol. I, S. 143-148, Berlin, 2000.

[Moving Averages page] Moving Averages page at StockCharts.com, http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:moving_averages

[Müller et al. 1997] Müller, K., Smola, A. J., Rätsch, G., Schölkopf, B., Kohlmorgen, J., and Vapnik, V. 1997. *Predicting Time Series with Support Vector Machines.* In Proceedings of the 7th international Conference on Artificial Neural Networks (October 08 - 10, 1997). W. Gerstner, A. Germond, M. Hasler, and J. Nicoud, Eds. Lecture Notes In Computer Science, vol. 1327. Springer-Verlag, London, pp. 999-1004.

[Nesreen et al. 2009] Nesreen K. Ahmed, Amir F. Atiya, Neamat El Gayar, and Hisham El-Shishiny, *An empirical comparison of machine learning models for time series forecasting,* accepted for publication, to appear in Econometric Reviews, 2009.

[Opiyo 1995] Opiyo E. T. O., 1995, *Case Based Reasoning for Expertise Relocation in Support of Rural Health Workers in Developing Countries*, Proceedings ICCBR-95, Lecture Notes in Artificial Intelligence 1010, Pages 77-87.

[Park 2004] Park, J. H., Im, K. H., Shin, C., and Park, S. C. 2004. MBNR: Case-Based Reasoning with Local Feature Weighting by Neural Network. *Applied Intelligence* 21, 3 (Nov. 2004), 265-276.

[Pavlidis and Horowitz 1974] Pavlidis, T., Horowitz, S. *Segmentation of plane curves.* IEEE Transactions on Computers; 1974 August; Vol. C-23(8), pp. 860-870.

[Plummer 2001] Eric A. Plummer, *TIME SERIES FORECASTING WITH FEED-FORWARD NEURAL NETWORKS: GUIDELINES AND LIMITATIONS*, Master Thesis, University of Wyoming, 2001.

[Popivanov et al. 2002] Popivanov, I., Miller, R. J. *Similarity search over time series data using wavelets.* In proceedings of the 18th Int'l Conference on Data Engineering; 2002 Feb 26-Mar 1; San Jose, CA, pp 212-221.

[Ratanamahatana and Keogh 2004] Ratanamahatana, C.A., Keogh, E. *Making Time-Series Classification More Accurate Using Learned Constrints.* Proceedings of SIAM International Conference on Data Mining; 2004 Apr 22-24; Lake Buena Vista, FL, pp. 1 l- 22.

[Ratanamahatana et al. 2005] Chotirat Ann Ratanamahatana, Jessica Lin, Dimitrios Gunopulos, Eamonn Keogh, Michail Vlachos, Gautam Das: *MINING TIME SERIES DATA, chapter in Data Mining and Knowledge Discovery Handbook*: Maimon, Oded; Rokach, Lior (Eds.), ISBN: 978-0-387-24435-8, Springer, 2005.

[Refenes 1995] Refenes, A.N., (1995) *Neural Networks in the Capital Markets.* John Wiley, Chichester.

[Reinartz 2000] Reinartz, T., Iglezakis, I., Roth-Bergofer, T., (2000), "On Quality Measures for Case Base Maintenance", *5th European Workshop (EWCBR 2000),* pp. 247-260, Trento, Italy, September 2000.

[Richter 1995] Richter M. M., 1995, *The Knowledge Contained in Similarity Measures,* Invited talk at ICCBR-95.

[Ruiz-Suarez et al. 1995] Ruiz-Suarez, J.C., Mayora-Ibarra, O.A., Torres-Jimenez, J., Ruiz-Suarez, L.G., (1995) *Short-term ozone forecasting by artificial neural networks.* Advances in Engineering Software 23, pp. 143–149.

[Schwartz 1997] Schwartz, A. B., Barcia R. M., Martins A., Weber Lee R. 1997, *PSIQ – A CBR Approach to the Mental Health Area*, 5th German Workshop on CBR – Foundations, Systems and Applications, Report LSA-97-01E, Pages 217-224.

[Shahabi et al. 2000] Shahabi, C., Tian, X., Zhao, W. *TSA-tree: a wavelet based approach to improve the efficiency of multi-level surprise and trend queries.* In proceedings of the 12th Int'l Conference on Scientific and Statistical Database Management; 2000 Jul26-28; Berlin, Germany, pp 55-68.

[Simić et al. 2003] Simić, D., Kurbalija, V., Budimac, Z.: *An Application of Case-Based Reasoning in Multidimensional Database Architecture.* In Proc. Of 5th International Conference on Data Warehousing and Knowledge Discovery (DaWaK), Lecture Notes in Computer Science, Vol. 2737. Springer-Verlag, Berlin Heidelberg New York (2003) 66 - 75.

[Simić et al. 2005] Simić D., Budimac Z., Kurbalija V., Ivanović M., *Case-Based Reasoning for Financial Prediction*, Lecture Notes in Artificial Intelligence (LNAI), ISSN: 0302-9743, vol. 3533/2005, pp. 839-842, (Proceedings of 18th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, IEA/AIE 2005, Bari, Italy, June 22-24 2005).

[Srinivasan et al. 1994] Srinivasan, D., Liew, A.C., Chang, C.S., (1994) *A neural network short-term load forecaster.* Electric Power Systems Research 28, pp. 227–234.

[Sykes Web] Sykes, A.O. *An Introduction to Regression Analysis* (Inaugural Course Lecture), http://www.law.uchicago.edu/Lawecon/WkngPprs_01-25/20.Sykes.Regression.pdf

[Tay and Cao 2001] Tay FEH, Cao LJ. *Application of support vector machines in financial time series forecasting.* Omega 2001; 29. pp. 309–317.

[Tay and Cao 2002] Tay FEH, Cao LJ. *Modified support vector machines in financial time series forecasting.* Neurocomputing 2002; 48. pp. 847–861.

[Tufte 1983] Tufte, E. *The visual display of quantitative information.* Graphics Press,Cheshire, Connecticut, 1983.

[Turchin et al. 2006] Turchin P., Grinin L., Munck V. C. de, Korotayev A. *History and mathematics: Historical Dynamics and Development of Complex Societies*. URSS Moscow (2006), ISBN-10: 5484010020.

[Turner 1995] Turner R.M. (1995). Orca: Intelligent adaptive reasoning for autonomous underwater vehicle control. In Proc. of the FLAIRS95 International Workshop on Intelligent Adaptive Systems, page 5262.

[University of Kaiserslautern, CBR home page] University of Kaiserslautern, Case-Based Reasoning homepage: http://www.cbr-web.org/

[Varma 1999] Varma, A. 1999. *ICARUS: Design and Deployment of a Case-Based Reasoning System for Locomotive Diagnostics*. In Proceedings of the Third international Conference on Case-Based Reasoning and Development (July 27 - 30, 1999). K. Althoff, R. Bergmann, and K. Branting, Eds. Lecture Notes In Computer Science, vol. 1650. Springer-Verlag, London, 581-595.

[Weber et al. 2000] Weber, M., lexa, M., Muller, W. *Visualizing Time Series on Spirals*. Proceedings of IEEE Symposium on Information Visualization; 2000 Oct 21-26; San Diego, CA, pp. 7-14.

[Weisberg 2005] Sanford Weisberg, *Applied Linear Regression*, Wiley-IEEE, 2005, ISBN 0471704083, 9780471704089.

[Wendler 2001] Jan Wendler, Steffen Brüggert, Hans-Dieter Burkhard, Helmut Myritz, *Fault-Tolerant Self-Localization by Case-Based Reasoning*, Lecture Notes in Computer Science, Volume 2019, Jan 2001, Page 259

[Wijk and Selow 1999] Wijk, J.J. van, E. van Selow. *Cluster and calendar-based visualization of time series data*. Proceedings of IEEE Symposium on Information Visualization; 1999 Oct 25-26, IEEE Computer Society, pp 4-9.

[Wikipedia] Wikipedia, The Free Encyclopedia, http://en.wikipedia.org/

[Wolfram Mathworld] Wolfram Mathworld, http://mathworld.wolfram.com/ .

[Wu et al. 2000] Wu, L., Faloutsos, C., Sycara, K. and Payne, T. R. *FALCON: Feedback Adaptive Loop for Content-Based Retrieval*. In: Very Large Databases 2000, pp 297-306.

[Yi and Faloutsos 2000] Yi, B., Faloutsos, C. *Fast time sequence indexing for arbitrary Lp norms*. Proceedings of the 26th Int'l Conference on Very Large Databases; 2000 Sep 10-14; Cairo, Egypt, pp 385-394.

[Zehraoui et al. 2003] F. Zehraoui, R. Kanawati, and S. Salotti. *Case base maintenance for improving prediction quality*. In The 6th International Conference on Case-Based Reasoning (ICCBR-2003), pages 703–717, Trondheim, Norway, June 2003.

[Zehraoui et al. 2004] F. Zehraoui, R. Kanawati, S. Salotti. *CASEP2: Hybrid case-based reasoning system for sequence processing*, European Conference on Case Based Reasoning (ECCBR 2004), Madrid, Spain, Lecture Notes in Computer Science, Volume 3155/2004, 2004, pp 449-463.

[Zhang et al. 1998] Guoqiang Zhang, B. Eddy Patuwo, Michael Y. Hu, *Forecasting with artificial neural networks: The state of the art*, International Journal of Forecasting, Volume 14, Issue 1, 1 March 1998, pp. 35-62.

INDEX OF FIGURES AND TABLES

INDEKS SLIKA I TABELA

BIOGRAPHY

Vladimir Kurbalija was born on 12th May 1977, in Novi Sad. In the year 1992, he enrolled in the gymnasium "Jovan Jovanović Zmaj". After gymnasium he enrolled at the Faculty of Science, University of Novi Sad, Department of Mathematics and Informatics. He graduated in the year 2000, with the average grade 9.88 (max. 10). On October 23, 2000 he defended his graduation thesis *Realization of Lexical and Syntax Analyzer for Programming Language Tiger* with the mark 10.

He is the winner of several faculty and university prizes for extraordinary success in the first, second and fourth year of studies, and also for his success during the whole studies. Also, he is the winner of the university prize named *Aleksandar Saša Popović* for the year 2000. (for the best graduation thesis in the area of Computer Science for the current year), and for the year 2002. (for exceptional scientific paper in the area of Computer Science).

He has been employed as the assistant at the Department of Mathematics and Informatics since 2000. He conducts exercises for the courses in *Internet Tools, Web Design, Compiler Construction and Data Structures and Algorithms* for the students of computer

science. During the same year he enrolled in post-graduate studies at the Department of Mathematics and Informatics of the Novi Sad University. On December 12, 2006 he defended his Master thesis *Implementation of Decision Support Systems Using Case Based Reasoning Technology*.

He was a secretary and a member of organizing committees of several seminars and conferences. From May 2009, he is Editorial assistant of the journal *Computer Science and Information Systems*. He also participated in these projects:

- Development of (intelligent) techniques based on software agents for application in information retrieval and workflow, 2002 – 2004. Supported by Ministry of Science and Technologies (Republic of Serbia), project no. 1844.
- Abstract Methods and Applications in Computer Science, Project no. 144017A Ministry of Science and Technologies (Republic of Serbia), 2006-2010.

During the time from September 2002 until January 2003, he stayed at the Humboldt University, Berlin, Research Group Artificial Intelligence, as a participant of the project *"Utilization of Case-Based Reasoning Technology for Implementation of Decision Support Systems"*. Furthermore, in June 2005, he stayed for one month in Linz, Austria at Johannes Kepler University, Institute for System Software as a participant of the project *"Innovation of Compiler Construction Course"*.

He is the author or a co-author of 13 scientific papers, of which 10 are published internationally. His fields of interests are: Artificial Intelligence, Data Mining, Case-Based Reasoning and Time series analysis.

BIOGRAFIJA

Vladimir Kurbalija je rođen 12. 05. 1977. u Novom Sadu. 1992 upisuje Gimnaziju "Jovan Jovanović Zmaj", prirodno-matematički smer. Posle završene gimnazije 1996. godine upisuje se na Prirodno-matematički fakultet u Novom Sadu, na smer diplomirani informatičar (Departman za matematiku i informatiku), gde je i diplomirao 2000. godine sa prosečnom ocenom 9,88. Diplomski rad pod nazivom *Realizacija leksičkog i sintaksnog analizatora za programski jezik Tiger* odbranio je 23.10.2000. sa ocenom 10.

Dobitnik je univerzitetskih i fakultetskih nagrada za izuzetan uspeh postignut u prvoj, drugoj i četvrtoj godini studija, kao i za uspeh postugnut u toku celokupnih studija. Takođe je dobitnik univerzitetske nagrade *Aleksandar Saša Popović* za 2000. godinu (za najbolji diplomski rad iz oblasti računarskih nauka u tekućoj godini) i za 2002 godinu (za izuzetan naučni rad u oblasti računarskih nauka).

Na Departmanu za matematiku i informatiku Prirodno-matematičkog fakulteta u Novom Sadu radi od 2000. godine kao asistent-pripravnik, gde drži vežbe iz predmeta *Internet alati, Web dizajn, Konstrukcija kompajlera* i *Strukture podataka i algoritmi*. Iste godine upisao je magistarske studije na Prirodno-matematičkom fakultetu u Novom Sadu. Magistarsku tezu sa nazivom *Implementacija sistema za podršku odlučivanju korišćenjem „Case Based*

Reasoning" tehnologije odbranio je 12. decembra 2006. godine i stekao akademski naziv magistra računarskih nauka.

Bio je sekretar i član organizacionih odbora nekoliko konferencija u zemlji. Od maja 2009. angažovan je i kao asistent editora časopisa *Computer Science and Information Systems.* Učešće na projektima:

- Razvoj (inteligentnih) tehnika zasnovanih na softverskim agentima za primenu u pretraživanju podataka i tokovima poslova, Ministarstvo za nauku, tehnologije i razvoj Republike Srbije – projekat broj 1844, 2002-2004.
- Apstraktni metodi i primene u računarskim naukama, Projekt 144017A Ministarstva nauke i zaštite životne sredine Republike Srbije (2006-2010)

U periodu septembar 2002 – januar 2003 boravio je na Humboldt Univerzitetu u Berlinu, grupa za veštačku inteligenciju kao učesnik na projektu *"Utilization of Case-Based Reasoning Technology for implementation Decision Support Systems"*. Takođe, mesec dana (jun 2005) je boravio na Johannes Kepler Univerzitetu u Lincu, Institut za sistemski softver kao učesnik na projektu *"Innovation of Compiler Construction Course"*.

Autor ili koautor je 13 naučnih radova, od kojih su 10 objavljenih u inostranstvu. Bavi se veštačkom inteligencijom, Data Mining-om, zaključivanjem na osnovu slučajeva (eng. Case-Based Reasoning) i analizom vremenskih serija.

UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI FAKULTET
KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj: RBR	
Identifikacioni broj: IBR	
Tip dokumentacije: TD	Monografska dokumentacija
Tip zapisa: TZ	Tekstualni štampani materijal
Vrsta rada: VR	Doktorska disertacija
Autor: AU	Vladimir Kurbalija
Mentor: MN	Mirjana Ivanović
Naslov rada: MR	Analiza i predviđanje toka vremenskih serija pomoću “Case-Based Reasoning” tehnologije.
Jezik publikacije: JP	Engleski
Jezik izvoda: JI	Srpski / Engleski
Zemlja publikovanja: ZP	Srbija
Uže geografsko područje: UGP	Vojvodina
Godina: GO	2009
Izdavač: IZ	Autorski reprint
Mesto i adresa: MA	Novi Sad, Trg Dositeja Obradovića 4
Fizički opis rada: FO	(7, 142, 118, 4, 33, 0, 0)
Naučna oblast: NO	Računarske nauke
Naučna disciplina: ND	Veštačka inteligencija
Predmetna odrednica / Ključne reči: PO	Case-Based Reasoning, Analiza vremenskih serija, Predviđanje vremenskih serija

UDK:
čuva se:
ČU

Važna napomena:
VN

Nema

Izvod:
IZ

U ovoj doktorskoj disertaciji prikazan je interesantan i perspektivan pristup rešavanja problema analize i predviđanja vremenskih serija korišćenjem Case Based Reasoning (CBR) tehnologije. Detaljno su opisane osnove i glavni koncepti ove tehnologije. Takođe, data je komparativna analiza različitih pristupa u analizi vremenskih serija sa posebnim osvrtom na predviđanje. Kao najveći doprinos ove disertacije, prikazan je sistem *CuBaGe* (Curve Base Generator) u kome je realizovan originalni način reprezentacije vremenskih serija zajedno sa, takođe originalnom, odgovarajućom merom sličnosti. Robusnost i generalnost sistema ilustrovana je realnom primenom u domenu finansijskog predviđanja, gde je pokazano da sistem jednako dobro funkcioniše sa standardnim, ali i sa nekim nestandardnim vremenskim serijama (neodređenim, retkim i neekvidistantnim).

Datum prihvatanja teme od strane NN veća:
DP

Datum odbrane:
DO

Članovi komisije:
KO

Predsednik:	dr Zoran Budimac, redovni profesor, Prirodno-matematički fakultet, Novi Sad
Član:	dr Mirjana Ivanović, redovni profesor, Prirodno-matematički fakultet, Novi Sad
Član:	dr Miloš Racković, redovni profesor, Prirodno-matematički fakultet, Novi Sad
Član:	dr Ivan Luković, redovni profesor, Fakultet tehničkih nauka, Novi Sad

UNIVERSITY OF NOVI SAD
FACULTY OF SCIENCE
KEY WORDS DOCUMENTATION

Accession number: ANO	
Identification number: INO	
Document type: DT	Monograph type
Type of record: TR	Printed text
Contents Code: CC	PhD Thesis
Author: AU	Vladimir Kurbalija
Mentor: MN	Mirjana Ivanović
Title: TI	Time Series Analysis and Prediction Using Case-Based Reasoning Technology
Language of text: LT	English
Language of abstract: LA	Serbian / English
Country of publication: CP	Serbia
Locality of publication: LP	Vojvodina
Publication year: PY	2009
Publisher: PU	Author's reprint
Publ. place: PP	Novi Sad, Trg Dositeja Obradovića 4
Physical description: PD	(7, 142, 118, 4, 33, 0, 0)
Scientific field: SF	Computer Science
Scientific discipline: SD	Artificial Intelligence
Subject / Key words: SKW	Case-Based Reasoning, Time Series Analysis, Time Series Forecasting

UC:
holding data:
HD

Note:
NO

None

Abstract:
AB

This thesis describes one promising approach where a problem of time series analysis and prediction was solved by using Case Based Reasoning (CBR) technology. Foundations and main concepts of this technology are described in detail. Furthermore, a detailed study of different approaches in time series analysis is given. System *CuBaGe* (Curve Base Generator) - A robust and general architecture for curve representation and indexing time series databases, based on Case based reasoning technology, was developed. Also, a corresponding similarity measure was modelled for a given kind of curve representation. The presented architecture may be employed equally well not only in conventional time series (where all values are known), but also in some non-standard time series (sparse, vague, non-equidistant). Dealing with the non-standard time series is the highest advantage of the presented architecture.

Accepted by the Scientific Board on:
ASB

Defended:
DE

Thesis defend board:
DB

President:	dr Zoran Budimac, full professor, Faculty of Science, Novi Sad
Member:	dr Mirjana Ivanović, full professor, Faculty of Science, Novi Sad
Member:	dr Miloš Racković, full professor, Faculty of Science, Novi Sad
Member:	dr Ivan Luković, full professor, Faculty of Technical Sciences, Novi Sad