Doktorandský den ’03

Ústav informatiky

Akademie věd České republiky

Paseky nad Jizerou, 25. – 26. září 2003

Obsah

Doktorandský den Ústavu informatiky Akademie věd České republiky se koná již po osmé, nepřetržitě od roku 1996. Tento seminář poskytuje doktorandům, podílejícím se na odborných aktivitách Ústavu informatiky prezentační možnosti pro výsledky jejich odborného studia a následného směrování jejich výzkumu. Současně poskytuje prostor pro oponentní připomínky k přednášené tematice a použité metodologii práce, ze strany přítomné odborné komunity.

Z jiného úhlu pohledu, toto setkání doktorandů podává průřezovou informaci o odborném rozsahu výzkumu, který je realizován na pracovištích či za spoluúčasti Ústavu informatiky AV ČR.

Od počátku organizování doktorandského dne byl soubor přednesených příspěvků publikován formou technických zpráv ÚI, které jsou dostupné elektronicky na adrese `www.cs.cas.cz`. Tento ročník se poprvé přistoupilo k vydání příspěvků v knižní formě. Jednotlivé příspěvky v publikaci jsou uspořádány adresně podle autorů, neboť uspořádání podle tematického zaměření nepovažujeme za účelné, vzhledem k rozmanitosti jednotlivých témat.

Vedení ÚI a vědecká rada ÚI jakožto organizátor tohoto odborného setkání věří, že toto setkání mladých doktorandů, jejich školitelů a ostatní odborné veřejnosti povede nutně ke zkvalitnění celého procesu doktorandského studia zajišťovaného v součinnosti s Ústavem informatiky, k odborné a pedagogické sebereflexi jak na straně doktorandů tak i na straně školitelů, a v neposlední řadě k navázání a vyhledání nových odborných kontaktů.

František Hakl
editor
1. září 2003

Current Trends in Middleware Benchmarking

doktorand:
LUBOMÍR BULEJ
Charles University in Prague
Faculty of Mathematics and Physics
Malostranske náměstí 25
Prague, Czech Republic

Institute of Computer Science
Academy of Sciences of the Czech Republic
Pod Vodárenskou věží 2
Prague, Czech Republic

lubomir.bulej@mff.cuni.cz
bulej@cs.cas.cz

školitel:
PETR TŮMA
Charles University in Prague
Faculty of Mathematics and Physics
Malostranske nám. 25
Prague, Czech Republic

petr.tuma@mff.cuni.cz

obor studia:
I2 – Softwarové systémy

Abstract

The paper provides a brief overview of the state of the art in middleware benchmarking. The overview is used to highlight some of the outstanding issues in the area, specifically the issues related to implementing, running and evaluating regression benchmarks. The issues are analysed in depth and the course of our work towards resolving them is presented.

1. Introduction

Middleware benchmarking is an integral part of the distributed computing area, where it serves to satisfy an obvious need to evaluate and compare performance of numerous implementations of communication and application middleware standards, such as CORBA [1], RMI [2], or EJB [3].

Benchmark results are regularly used both by the middleware users, to compare performance and functionality of software products from different vendors, running on different hardware and software platforms, and by the middleware developers, to evaluate the performance of their product in critical areas, to assess implementation changes that may have an impact on performance, and to determine the extent of such impact. In addition, benchmarking can be also used for monitoring and diagnostics, and in less common cases, for estimation of middleware performance under different, previously untested, conditions [4].

2. State of the Art

The area of middleware benchmarking currently lacks a standard that would unify the process of middleware benchmarking in general, so that it would cover benchmarking of different middleware platforms and provide a common methodology for obtaining the data and guidelines for presentation of the results. For specific middleware platforms, the situation is better due to demand for comparable and easy to understand results.

The most advanced in this regard appears to be the area of CORBA benchmarking, mostly through the efforts of OMG and its White Paper on Benchmarking [5], which standardizes the terminology, methodology and outlines the basic requirements for open, easy to measure and understandable benchmarks. As for other middleware platforms, such as EJB, the standardization exists in the form of widely accepted benchmarks, which are used throughout the industry for performance evaluation and comparison.

At present, we are aware of two principal types of benchmarks being widely used in the area, termed *model-application benchmarks* and *feature-specific benchmarks*. These differ substantially in the methodology for obtaining the data and in the presentation of results, but their popularity suggests that both types of benchmarks have its use and audience. Depending on the expected type of results, the audience can be split into the group of middleware users and the group of middleware developers, as described in [6].

2.1. Model-application Benchmarks

Model-application benchmarks usually simulate a model application and yield a tuple of numeric values expressing the performance of the benchmark application. Often, the results consist of a pair of values, of which one represents the number of model-application specific operations per second achieved on a given hardware and software platform, while the other amounts to the cost of a single operation. This practice is consistent with that of the Transaction Processing Council (TPC) benchmarks for database systems.

While the results produced by such benchmarks are easy to understand, it is very difficult to make any assumptions about performance of a system under different workloads or for other application types, since benchmarks of this kind do not collect data concerning the lower-level application-independent functionality of the middleware platform used.

As an example of this particular benchmark type, consider the EJB platform, which does have an officially standardized benchmark suite for measuring performance and scalability, called ECperf [7]. The suite measures performance of a model application simulating real-world business problems with respect to product manufacturing, order, inventory, and supply chain management. The benchmark results in a pair of values representing the achieved number of operations per second and the cost of a single operation.

RUBiS [8] is another example of a model-application benchmark and is based on a prototype auction site, modeled after eBay [9]. The benchmark is meant to be used to evaluate application design patterns and performance scalability of application servers. SPECjbb2000 [10] is a benchmark emulating a 3-tier system with business logic engine in the middle tier, which is currently the most common type of server-side Java application. This enumeration is by no means complete, the intent was to provide only a few examples for illustration purposes.

2.2. Feature-specific Benchmarks

The other type of benchmarks is intended to provide very detailed coverage of individual aspects of given middleware platform, such as marshaling, dispatching, or transport efficiency, including aspects specific to a particular implementation of that platform, such as threading model or shared memory optimizations. The content of the results produced by such benchmarks usually consists of very detailed and technical data, and is expected to be read and understood by the developers of the particular middleware platform with appropriate background.

In measuring performance of isolated aspects typical for a specific middleware platform, these benchmarks provide information about behavior of the individual aspects on a specific hardware and software platform and under various situations, such as heavy thread usage or network distribution. Middleware exhibiting no anomalies in the behavior of its individual aspects is expected to be performing well in the overall scope.

An example of a feature-specific benchmarking suite is the Open CORBA Benchmarking Suite [6] for performance evaluation of CORBA middleware. In dealing with the issues mentioned in the White Paper on CORBA Benchmarking, such as uniform methodology, ease of use, and openness of the whole process from data collection to publication of the results, the suite serves as a reference for conducting CORBA benchmarks.

To our best knowledge, there is no comparable benchmark suite for other middleware platforms, especially RMI and EJB. As for CORBA, the distribution of TAO [11], an open-source implementation of the OMG CORBA specification, contains benchmarks measuring selected aspects of the CORBA ORB performance. The Open CORBA Benchmarking suite differs from the collection of TAO benchmarks in that it collects more detailed data and that it can be easily used for may different ORB implementations, because it is not tied to a particular ORB at the source code level.

3. Regression Benchmarking

Benchmarks can be very helpful during middleware development. Our experience from a series of CORBA performance evaluation and comparison projects shows that systematic benchmarking of middleware primitives can reveal performance bottlenecks and bad design decisions as well as implemetation errors. In other words, detailed, extensive and repetitive benchmarking can be used for finding regressions in middleware implementations, hence the term *regression benchmarking*.

Even though it is beyond doubt that regression benchmarking is a valuable tool in the course of middleware development, our practical experience indicates it is rarely used, both in the commercial and the non-commercial/open-source domains. As for the open-source middleware implementations, the only exception from the fact appears to be the already mentioned TAO, which besides having a suite of regression tests to validate correct functionality, also has a number of benchmarks. The collection, however, does not support automation, which is needed to perform repeated extensive testing and performance evaluation.

Commenting on the development practices of commercial closed-source middleware vendors is more difficult, as their practices are closed and our knowledge is thus limited to our experience with their products. Nevertheless, in the past years we have revealed several performance problems in leading commercial ORB implementations, which ranged from minor flaws to major scalability issues. These problems would probably have been found had the software been subjected to regression benchmarking. From this we conclude, that regression benchmarking has not yet found a regular use even in the commercial sphere. The fact that most available benchmark results appear to have been edited and formatted by hand also indicates that the benchmarks are not conducted very often, which only supports our argument.

In our analysis of why the use of regression benchmarks is not more widespread, a major factor appears to be the fact that the initial cost of setting up regression benchmarking for a software project is perceived to be higher than the potential benefits. Therefore, there is little or no incentive to invest resources in it. Closer look at the initial costs reveals what we believe are the major issues causing the perception of unreasonably high costs:

1. *Creation of benchmarks*

   Creating regression benchmarks requires developer resources, which may not be readily available. An important factor is the amount of work required to benchmark a specific functionality on a given platform. Unless the effort/cost necessary to create and, which is more important, update and extend the regression benchmark suite is reasonably small, the developers and especially the managers are reluctant to invest effort into work, that does not provide immediate profit.

2. *Execution of benchmarks*

   Even when the necessary effort has been invested into creation of a suite for regression benchmarking, our experience shows that it is non-trivial to conduct the actual measurements and conduct them repeatedly, which is essential for regression benchmarking. The benchmark suite can contain hundreds of tests and take days or weeks of running time to complete. The amount of collected data can easily reach the order of gigabytes. Without a mechanism for automated execution and collection of benchmark data, the task is on the verge of possibility. Also, for the benchmarks to be conducted on daily basis, the execution time of the entire suite becomes an issue and requires a control mechanism.

3. *Evaluation of benchmarks*

   The data collected during automated execution of the regression benchmarking suite are mostly raw numbers and in amounts, that prevent direct analysis by a human. The data must be first processed by a machine and the size and detail reduced to a manageable level. It should be possible to evaluate the data automatically, at least to identify anomalies and to alert the developers to pay extra attention to the results. As in the previous case, without automated processing facilities, the benchmark results are merely gigabytes of useless data.

Based on our experience with industrial partners, we have created a regression benchmark suite, which attempts to address the above issues using the following concepts:

3.1. Benchmarking Framework

The solution to the issue (1) above requires easy extensibility of a suite in form of new benchmarks, while a part of the solution to issue (2) should handle automated benchmark execution and collection of the results. To address these issues, we have created a benchmarking framework with focus on the following features.

Easy creation of new benchmarks

To be of any use to a middleware developer, the benchmarking framework must not constitute an obstacle. If there is a need to benchmark a specific middleware functionality, it is enough to write the code to test the functionality and reuse the generic facilities provided by the framework, such as support for data acquisition, configuration, multi-threading, variation of test parameters, etc.

Portability and extensibility of the benchmark suite

If a particular middleware supports multiple platforms, it is desirable to compare the performance of the middleware on those platforms. Our framework is reasonably portable to commonly used software and hardware platforms, including several operating systems, compilers, different version of the middleware implementation and vendor-specific tools. Extending the suite to support a new broker is a matter of minutes, adding support for a completely new platform takes longer because of the platform dependent code which has to be supplied to the framework.

Automated compilation and unattended execution

Manual compilation, configuration and execution of suite benchmarks, as is the case of some application oriented benchmarks for EJB servers is simply not an option. Our suite contains hundreds of individual benchmarks and supports running them either locally, or remotely on two different nodes on the network. The Cygwin [12] environment is used to support remote execution of benchmarks on machines running the Windows NT class of operating systems.

3.2. Sample Collection Mechanism

Since we want a regression benchmark to produce very detailed information, we cannot settle for collecting only averages, minima, and maxima of the observed values. Very often, the distribution of the samples, its median and inter-quartile range are much more telling that the averages, especially if we are interested in real time behavior or in comparing performance of a particular middleware implementation on different platforms. Therefore, in addition to average, minimal, and maximal values, our framework also collects individual samples for a selected thread of execution.

In our framework, the execution of a benchmark progresses through several stages and the amount of time required for a single benchmark run depends on the time spent in the execution of individual stages. The first stage is warm up, the second is collection of resource usage data both on the client and on the server, the third is collection of individual samples from a selected thread of execution, and the third is collection of averaged data.

To avoid interference caused by just-in-time compilation, disk cache flushes, stabilization of adaptive resource allocation algorithms and other phenomena accompanying application start up, the framework throws away data from the *warm up stage*, a fixed time interval at the beginning of the measurement.

Automated adjustment of benchmark execution time

With the exception of the third stage, all stages have a fixed execution time. The execution time of third stage is determined by a fixed number of individual samples to be collected and the complexity of the measured action. As such, its duration is not really known in advance and has a major impact on the total execution time of a single benchmark.

The execution times of individual stages and the number of samples to collect in the third stage have been chosen conservatively, so that we collect more than enough samples to obtain accurate data for further processing. Some of the values (such as the duration of the warm up stage) are probably very pessimistic, resulting in longer execution times than necessary. On the other hand, we prefer accurate data (and longer execution) to inaccurate data. Currently, the amount of data gathered from execution of the complete regression benchmark suite ranks in the order of gigabytes and its running time is measured in days or weeks.

Such running times are not so disastrous for a one-time extensive performance evaluation of particular middleware implementation, but are hardly acceptable for day-to-day measurements, which are of interest to the developers. To reduce the running time of the benchmark, it is necessary to be able to correctly determine the appropriate duration of the fixed-length stages as well as control the amount of data gathered in the variable-length stage without compromising the credibility of the data.

We expect to be able to control the execution time of the variable-length stage by specifying the required accuracy of the collected data. By being able to collect the right amount of individual samples to satisfy the accuracy requirement, we could control the execution time of the most unpredictable (in terms of execution time) stage of the benchmark, which would in turn produce significant savings in the overall execution time. As a bonus, we would need to collect and process less data. Besides determining the number of samples that need to be collected, we would like the framework to be also able to determine the appropriate length of the warm up stage.

Depending on the type of an individual benchmark, the observed values can be modeled by response variable with a single constant or variable factor and a random error. This fact led us to evaluate application of statistical methods to process the data and for estimation/control of benchmark timing parameters. Assuming that we can find appropriate statistical methods to estimate the length of the warm up stage and the number of samples required for the specified accuracy of the data, there is one more issue: outliers.

The observed data will contain outliers caused by unpredictable and hardly avoidable events, such as process rescheduling and other phenomena caused by the operating system interference. Such outliers should not be hidden from the developer so that it is possible to observe the real behavior of the middleware on a particular platform. On the other hand, the outliers will be potentially harmful for statistical methods that make use of variance σ^2 or sample variance s^2, both of which are very sensitive to extreme values in the population sample. We therefore need a mechanism to detect the outliers and exclude them from the benchmark parameter estimation/control process.

Potentially applicable statistical methods

The following section presents an overview of the statistical methods we consider suitable for solving the issues mentioned in the description of the sample collection mechanism above, along with a short description of expected application and planned experiments. Most of the methods come from the area of sequential statistical analysis, which is quite popular scientific discipline with many contributions from the area of discrete-event simulations.

Whether we can really benefit from the research done in that area is still subject to future work though, mainly to confirm that certain approaches used in discrete-event simulations can be used for middleware benchmarking. The values observed by the benchmark can be considered random, but except for a few specific cases, we cannot make any assumptions about the distribution of those values. The results from the simulation community mostly apply to stationary processes, i.e. processes whose mean, variance and autocorrelation structure do not change over time.

- *Outlier detection and removal*

  We plan to use outlier detection methods to identify and hide outliers from the sequential estimation methods that operate with sample variance s^2.

  – A simple method that can be used to detect outliers is the application of the Chebychev inequality and sample mean absolute deviation as an estimator of population standard deviation. Given the nature of outliers in our case, this method may be too pessimistic even for our purposes and throw away valid data points.

  – More complex procedure to identify the outliers is based on the analysis of the distance to an example's nearest neighbours. Knorr [13] defines distance based outlier as follows: An object O in a dataset T is a $DB(p, D) - outlier$ if at least fraction p of the objects in T lies farther than distance D from O. We either need to find the right values of the p and D parameters empirically, or find a way to determine them automatically for the method to give us the desired results.

- *Quantile and histogram estimation*

  We may want to evaluate using methods for quantile and histogram estimation instead of the methods for sequential mean estimation to determine the appropriate number of samples for predefined accuracy. Such methods should be considerably less susceptible to enormous deviations caused by the the presence of outliers.

  – A simple method for obtaining confidence interval for a population quantile is to use the standard non-parametric estimation based on the order statistics. While this methods may be sufficient for our purposes, we may also need to evaluate and experiment with other methods.

  – In [14] the authors discuss an implementation of a sequential procedure for constructing proportional half-width confidence intervals for simulation estimator of the steady-state quantiles and a histogram of stochastic processes.

  – Chen [15] describes an implementation of a two-stage procedure for constructing confidence intervals for a simulation estimator of the steady-state quantiles of stochastic processes. The algorithm dynamically increases the sample size so that the quantile estimates satisfy the proportional precision at the first phase and the relative or absolute precision at the second phase.

- *Warm up stage length estimation*

  This is a non-trivial task to solve automatically and we will probably have to use an empirical method with a mechanism for platform dependent parametrization. There is a number of methods for estimation of the length of the warm up stage in the simulation area, ranging from graphical methods, through heuristic approaches and statistic methods to various hybrid methods. Contrary to our case though, the problem is well defined in the area of discrete-efvent simulations in that it usually estimates the length of warm up stage of a stochastic process, which can be formally described and parametrized. For our purposes, would like to estimate the time necessary to filter out phenomena accompanying an execution of a program in an operating system, i.e. priming of processor caches, disk cache flushes, just in time compilation, all of which is rather hard to formalize.

- *Required number of samples estimation*

  The basic idea is to specify the relative precision of the mean and determine the number of samples needed to ensure that the mean will be within the confidence interval with probability $(1-\alpha)$, α being the significance level. Unfortunately, the straightforward two-stage procedure suggested by Stein [16] cannot be used, because the observed values do not have the normal distribution.

  Assuming that the random error component of the observed values comes from an arbitrary distribution with a mean value μ and a finite variance σ^2, according to the central limit theorem we can assume the sample mean to have approximately normal distribution with the same parameters as the original distribution. Therefore we may very well group the observed values, compute the sample means for individual groups and thus obtain a sequence of sample means, which is in fact a sequence of random variables having a normal distribution. This assumption is also used in a class of sequential methods called *batch means procedures*. These procedures use the grand mean of batch means as the estimator of the mean value of the population distribution. Confidence interval is then computed using the sample variance s^2 of the batch means and quantiles of Student's t-distribution.

  – Nakayama [17] describes a modification of Stein's two-stage procedure for use with the method of batch means. This is the primary method of interest for determining the minimal number of samples, but our preliminary testing shows that in presence of extreme outliers, which is our case as mentioned above, the estimation requires too many samples to be practical.

    Combined with removal of the extreme outliers the method produces fairly practical results, the remaining problem being the removal of outliers itself. In future work, we intend to focus on determining sound and defensible rules for removing outliers in order to obtain useful results from the sequential analysis process.

  – Hlavka [18] describes a three-stage procedure, which improves the asymptotic behavior of Stein's two-stage procedure by adding an additional sampling stage. Since this method cannot be used directly for the same reasons as the original Stein's two-stage method, we may want to determine whether it is possible to modify this method for use with the method of batch means. If the modification is possible, the method may give better results than the two-stage procedure described by Nakayama.

  – Steiger [19] summarizes the results of an extensive experimental performance evaluation of selected batch means procedures called ABATCH, LBATCH and ASAP. These methods are iterative and their operation may depend on the results of additional tests performed on the sequence of batch means, such as von Neumann test for independence or Shapiro-Wilk test for multivariate normality.

    This is the case of the ASAP method [20], which can adaptively change the batch size and the number of batches in response to the results of the additional tests. While the application of such methods seems attractive, we would prefer to achieve our goals using simpler (and less computationally intensive) methods so that we do not disturb the measurement by additional load caused by complex computations. We plan to evaluate the iterative methods in case the simpler methods fail.

3.3. Result Processing Mechanism

The amount and the nature of the data collected from the execution of the whole benchmark suite practically rule out any form of human processing of the collected data. To solve issue (3) mentioned above, automation is a must to reduce the amount of data to a level manageable by humans.

Automated processing and evaluation of the results

Even if we have optimal estimate of the number of samples that need to be collected in the variable-length stage of the benchmark, we still have to process significant amount of data. Currently, our framework is able to automatically generate HTML reports with dozens of images, which are much more suitable for human inspection than the raw data. The automatic evaluation of results is left as a topic for future work, where we expect to be able to detect anomalies in comparison with previous runs of the benchmark.

4. Conclusion

We have provided an overview of the state of the art in middleware benchmarking, which we have extended with the regression benchmarking approach. To do so, we have pointed out some of the major outstanding issues related to regression benchmarking and sketched the course in which we plan to address the issues in our future work.

To our best knowledge, we are not aware of related work comparable to our approach. Of course, there is a large number of various benchmarking projects for CORBA and EJB middleware, but none of them is currently comparable to our work either in the methodology used for conducting the measurements, or in the test coverage in case of CORBA benchmarking.

In the near future, we plan to perform practical experiments using statistical methods to estimate the minimal number of samples required to satisfy a predefined accuracy, methods for determining the appropriate length of the warm up stage as well as methods for detecting outliers in the population sample. Optional prototype results will be available at our website at http://nenya.ms.mff.cuni.cz.

Acknowledgements

We would like to express our thanks to František Plášil for his valuable comments.

References

[1] Object Management Group, *The Common Object Request Broker: Core Specification*, Dec 2002. OMG formal/02-12-02, version 3.0.2.

[2] Sun Microsystems, Inc., *Java Remote Method Invocation Specification*, 2002. Revision 1.8, Java2 SDK, version 1.4.

[3] Sun Microsystems, Inc., *Enterprise JavaBeans Specification*, Aug 2001. Version 2.0, Final Release.

[4] A. Buble and P. Tůma, "On Benchmarking Object-Oriented Middleware," in *Week of Doctoral Students Conference*, Faculty of Mathematics and Physics, Charles University, Prague, Jun 2000.

[5] Object Management Group, *White Paper on Benchmarking*, Dec 1999. OMG bench/99-12-01.

[6] P. Tůma and A. Buble, "Open CORBA Benchmarking," in *International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'01)*, (Orlando, FL, USA), SCS, Jul 2001.

[7] Sun Microsystems, Inc., *ECperf Specification*, Apr 2002. Version 1.1, Final Release.

[8] ObjectWeb Consortium, *RUBiS: Rice University Bidding System*. http://rubis.objectweb.org.

[9] eBay, Inc., *The World's Online Marketplace*. http://www.ebay.com.

[10] Standard Performance Evaluation Corporation, *SPECjbb2000: Java Business Benchmark*. http://www.spec.org/jbb2000.

[11] Distributed Object Computing Group, *TAO: The ACE ORB*. http://www.cs.wustl.edu/ schmidt/TAO.html.

[12] Red Hat, Inc., *Cygwin: A Unix-like environment for Windows*. http://www.cygwin.com.

[13] E. M. Knorr, R. T. Ng, and V. Tucakov, "Distance-Based Outliers: Algorithms and Applications," *VLDB Journal: Very Large Data Bases*, vol. 8, no. 3-4, pp. 237–253, 2000.

[14] E. J. Chen and W. D. Kelton, "Quantile and Histogram Estimation," in *Winter Simulation Conference*, 2001.

[15] E. J. Chen, "Two-Phase Quantile Estimation," in *Winter Simulation Conference*, 2002.

[16] C. Stein, "A Two-Sample Test for a Linear Hypothesis Whose Power is Independent of the Variance," *Annals of Mathematical Statistics*, vol. 16, pp. 243–258, Sep 1945.

[17] M. K. Nakayama, "Two-Stage Stopping Procedures Based on Standardized Time Series," *Management Science*, vol. 40, pp. 1189–1206, 1994.

[18] Z. Hlavka, *Robust Sequential Methods*. PhD thesis, Faculty of Mathematics and Physics, Charles University, Prague, 2000.

[19] N. M. Steiger and J. R. Wilson, "Experimental Performance Evaluation of Batch Means Procedures for Simulation Output Analysis," in *Winter Simulation Conference*, pp. 627–636, 2000.

[20] N. M. Steiger and J. R. Wilson, "Improved Batching for Confidence Interval Construction in Steady-State Simulation," in *Winter Simulation Conference*, pp. 442–451, 1999.

Automating Connector Evolution

doktorand:
TOMÁŠ BUREŠ
Academy of Sciences of the Czech Republic
Institute of Computer Science
bures@cs.cas.cz

školitel:
FRANTIŠEK PLÁŠIL
Academy of Sciences of the Czech Republic
Institute of Computer Science
plasil@cs.cas.cz

obor studia:
I2 – Softwarové systémy

Abstract

A connector is a first-class entity representing communication between components. It usually encapsulates several tasks (e.g. marshaling, unmarshaling, synchronization, logging, etc.). In this paper we follow a work done on connectors in our group and show how to automatically select a "connector configuration" based on designer's decision (communication style and non-functional properties).

1. Introduction

Connectors emerged recently in component-based systems as a first-class entity representing communication between components. They encapsulate all communication related tasks (e.g. marshaling, unmarshaling, synchronization, logging, adaptation, etc.) allowing components to contain only a business logic. This leads to two main benefits: (1) components are easier to develop, and (2) moving communication related code to connectors, components are middleware independent (i.e. they can use e.g. RMI [1], CORBA [2], SunRPC [3], etc. without a change).

Communication related tasks contained in connectors code are almost always the same. They comprise marshaling, unmarshaling, synchronization, etc. The only actual difference between two connectors instances is that they mediate different type of data and that they can utilize different middleware. Thus, we believe that stating basic connectors properties [4] (communication style and non-functional properties) we can generate connectors' code automatically with respect to target platforms (deployment docks).

2. Background and goal of the paper

In this paper is the continuation of the work done on connectors in our group [5], which presents a connector model where a connector is build (similar to components) from smaller parts called elements (see Figure 1). The elements can be compound (composed of other elements) or primitive (directly coded in a programming language). The composition of elements forming a connector or a compound element is called connector resp. element architecture. Unlike components, connector elements (both compound and primitive) have generic interfaces, which are fixed, with respect to components being interconnected, only in one of the later phases of connector generation. (This process is called element adaptation.)

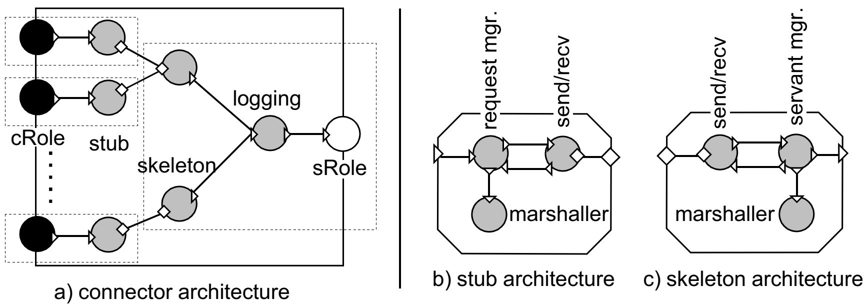


Figure 1: Example of a procedure call architecture

Basically the process of connector generation comprises several steps as depicted on Figure 2 (connector refinement tree). On the top (in the root of the tree) there is a designer's rough decision what a connector "should do". It is expressed in form of selection of a communication style (procedure call, messaging, streaming, blackboard) and non-functional properties [4]. The non-functional properties (NFPs) describe connector's behavior non-visible to attached components such as distribution, logging, etc.

In the second step an appropriate connector architecture is selected with respect to the prescribed communication style and NFPs. Obviously there can be several architectures implementing a particular communication style varying in a set of supported NFPs. (There can be e.g. a procedure call architecture with stubs and skeletons allowing for *distribution*, as well as an architecture without stubs and skeletons which would be faster but would allow only for calls within one address space.)

In the third step an implementation element is chosen for each element. Again, one element can have several possible implementations depending on values of NFPs (e.g. CORBA-based stubs and skeletons support the *distribution* property with value 'CORBA', RMI-based stubs and skeletons support it with value 'RMI', etc.). If a compound element is chosen for an element implementation, this step is recursively applied on its contents.

In the fourth step, generic roles on a connector are fixed (with respect to the attached components) and the interfaces are propagated inside the connector causing generic interfaces on elements' ports to be fixed too (aka element adaptation).

As the result of the fourth step, a complete connector configuration is obtained specifying a connector architecture, element implementations and fixing all generic interfaces. Now the connector generation is completed by adapting used element implementations to the interfaces prescribed by the fourth step. The connector is eventually created at runtime by instantiating single adapted element implementations and connecting them according to the connector architecture.

In our recent work we described how to perform the last step (element adaptation) [6] and how to instantiate a connector; provided that a complete connector configuration is specified. However, the path from a designer's decision (on the top) down to a complete connector configuration is difficult, requiring deep knowledge of all target platforms, available elements' implementations, etc. Thus, to simplify the connectors usage we need to perform the several steps automatically.

The goal of this paper is therefore to show how to perform the steps resulting into a complete connector configuration automatically.

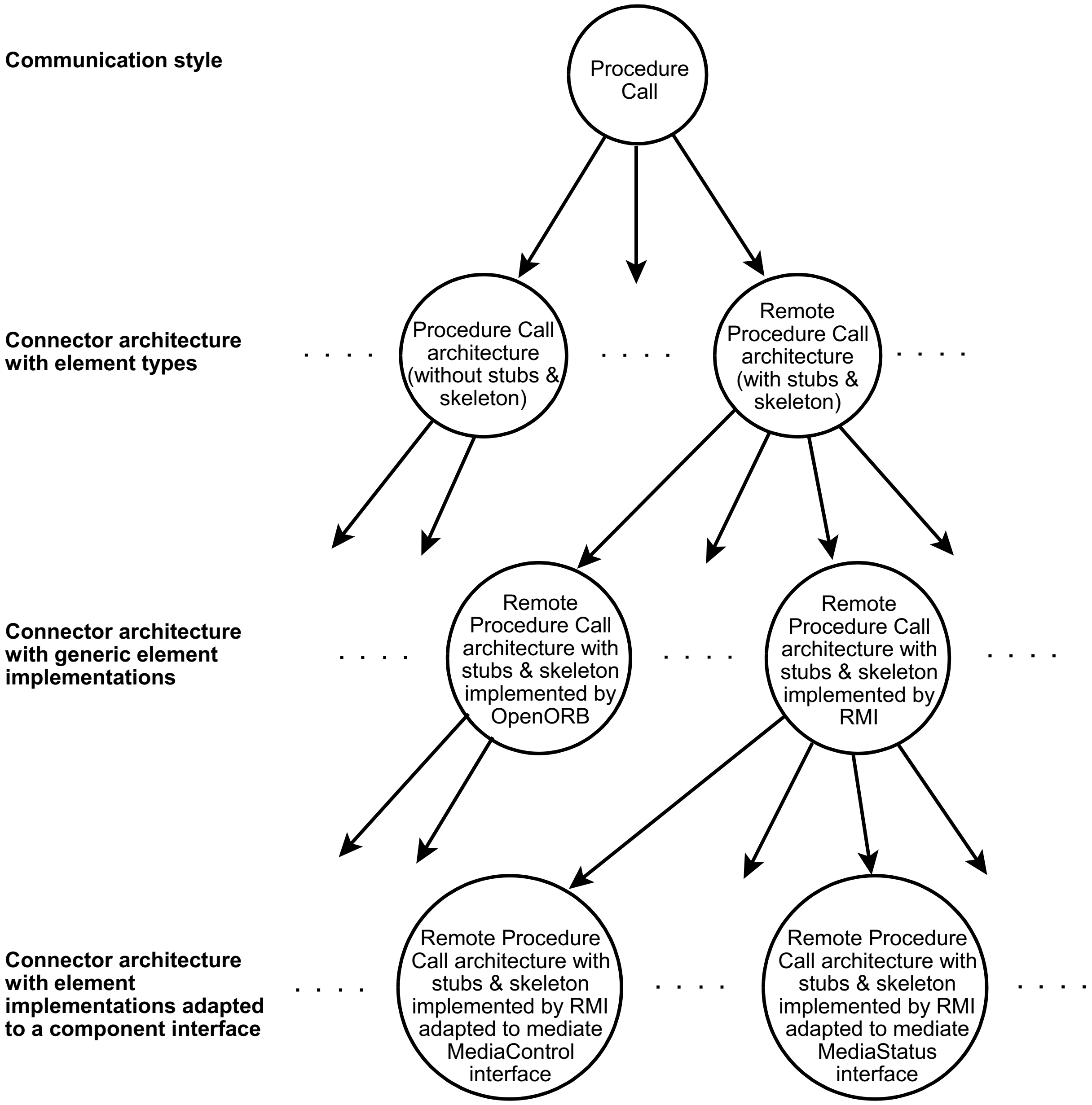


Figure 2: Connector refinement tree

3. Selecting a connector configuration

In this section we describe how to select a complete connector configuration based on requested communication style and NFPs. Having presented the refinement tree in the previous section, the problem of connector generation is reduced to finding a path in the tree from the top layer (designer's decision) to the bottom layer (complete connector configuration). Obviously, not all nodes in the tree are correct with respect to requested communication style and NFPs. However, these are not the only problems which can occur. Basically we recognize three basic kinds of inconsistencies:

1. *Composition inconsistency.* Selected element implementations are mutually incompatible (e.g. CORBA-based stub with RMI-based skeleton).

2. *NFP inconsistency.* A connector does not comply with all requested NFPs.

3. *Environment inconsistency.* An element implementation is incompatible with a target platform (e.g. necessary libraries are missing).

In the rest of the section we present logical predicates which assure that a particular inconsistency does not occur. Having these rules, a path in the refinement tree leading to a correct connector configuration can be obtained using backtracking.

3.1. Composition inconsistency

Composition consistency is checked by matching "types" on adjacent element ports. For simplicity reason, we show this on an example. Let us suppose the *request manager* element, which breaks a call into a sequence of a request and a response (see Figure 1). This element defines four ports (*call*, *mashall*, *lineIn*, *lineOut*) used to bind to neighboring elements. In ADL notation [6] this is defined in the following way:

```
element frame RequestManager {
  provides: call;
  requires: marshall;
  provides: lineIn;
  requires: lineOut;
}
```

We enrich every element implementation of a *request manager* by a relation between interfaces on its ports. In our example it will look this way:

$$I_{marshall} = marshall(Protocol, I_{call})$$
$$I_{lineOut} = request(Protocol, I_{call})$$
$$I_{lineIn} = response(Protocol, I_{call})$$

The variable $I_{marshall}$, I_{call}, etc. represent interfaces assigned to a particular port, the local variable $Protocol$ represents communication protocol (e.g. 'SunRPC'), and terms $marshall$, $request$ and $response$ denote a parametrized interface (e.g. $marshall(Protocol, I_{call})$ would contain methods $pack$ and $unpack$ which serialize methods from I_{call} using external data representations defined by $Protocol$). It is important to note that in our approach the word 'interface' does not indicate only a set of methods but denotes a use case and expected behavior as well.

Having associated every element implementation with such relation over ports, we state predicates that arise with every inter-element binding (both local and remote).

local binding: $I_{Elem1Port} = I_{Elem2Port}$
remote binding: $transport\_compatible\_bind(I_{Elem1Port}, I_{Elem2Port})$

The variables $I_{Elem1Port}$ and $I_{Elem2Port}$ denote interfaces on the interconnected ports. The predicate $transport\_compatible\_bind$ is defined with respect to available middleware. In our case, e.g.:

$$\begin{array}{l}\vdash transport\_compatible\_bind\_SunRPC(\\ \quad udp(request('SunRPC', I_{call}), response('SunRPC', I_{call})), \\ \quad udp(response('SunRPC', I_{call}, request('SunRCP', I_{call})))\end{array}$$

$$\begin{array}{l}\vdash transport\_compatible\_bind(I_{Elem1Port}, I_{Elem2Port}) \longleftrightarrow \\ \quad transport\_compatible\_bind\_SunRPC(I_{Elem1Port}, I_{Elem2Port}) \vee \\ \quad transport\_compatible\_bind\_RMI(I_{Elem1Port}, I_{Elem2Port}) \vee \\ \quad \ldots,\end{array}$$

where $udp(In, Out)$ denotes UDP-based communication.

As the result of element binding we get a relation between connector roles (component attachment points). A connector is then composition consistent if all port interface relation and binding predicates present in the connector configuration hold and interfaces of attached components are substituable to respective connector roles.

3.2. NFP inconsistency

NFP inconsistency is checked over a complete connector configuration. For each NFP a predicate exists that tests whether a given configuration satisfies a particular NFP value. These predicates are constructed using lower level predicates able to test NFP consistency for a particular element and architecture.

Let us clarify this approach on the example in Figure 1. There exists a predicate for testing the *distribution* property in the used connector architecture. It takes the following form:

$$\begin{array}{l}\vdash nfp\_mapping\_distrib\_RPCImpl('distribution', NFPValue, ConConfig) \longleftrightarrow \\ \quad (con\_arch\_name(ConConfig, 'RemoteProcCallImpl') \wedge \\ \quad con_elem(ConConfig, 'Stub') \wedge \\ \quad nfp_mapping('distribution', NFPValue, StubElemConfig))\end{array}$$

This predicate yields true if a connector is constructed using a particular architecture ('RemoteProcCallImpl') and delegates decision about the value of the *distribution* property to an implementation of the stub element. All stub element implementations are then associated with similar predicates, e.g.:

$$\begin{array}{l}\vdash nfp\_mapping\_distrib\_RMIStubImpl('distribution', 'RMI', ConConfig) \longleftrightarrow \\ \quad con_arch_name(ConConfig, 'RMIStubImpl')\end{array}$$

The stub element implementations define the value directly ('RMI' in our example) or delegate the decision to the marshaller element (in case of a compound stub).

All predicates for testing NFPs are finally put together to form one composed predicate used to test arbitrary NFP regardless which architecture and element implementations were selected:

$$\begin{array}{l}\vdash nfp\_mapping(NFPName, NFPVal, ConConfig) \longleftrightarrow \\ \quad nfp\_mapping\_distrib\_RPCImpl(NFPName, NFPVal, ConConfig) \wedge \\ \quad nfp\_mapping\_distrib\_RMIStubImpl(NFPName, NFPVal, ConConfig) \wedge \\ \quad \ldots,\end{array}$$

3.3. Environment inconsistency

In order to exclude element implementations which would not run propertly on a target platform, we enrich their definition by a set of environment requirements that have to be satisfied by a target platform. For the time being, we use for the requirement specification a simple set of key-value pairs (such as 'SunRPC' ⇒ '2'). Every deployment dock declares its provisions in form of key-value pairs too. The environment consistency is then assured if for each element implementation used in a connector holds that its set of environment requirements is a subset of provisions of a deployment dock where the element is going to be instantiated. This approach is very simplistic though, not allowing to express complex requirements. Thus, in future work we plan to enhance the mode by a more sophisticated requirement specification.

4. Evaluation and related work

To our knowledge there is no related work addressing the connector generation in this a complexity. Basically three areas of our work are also discussed elsewhere:

1. *Connector model.* There is a few component models supporting connectors. To the most important belong C2 [7], Acme [8], and Unicon [9]. These models lack several features though which we require from a decent connector modeli [10]. In C2 components communicate via an 'inteligent' bus capable of message filtering, but not supporting any other types of connectors. Acme on the other hand allows for fully user-defined connectors, however, as a tool for prototyping and reasoning about component systems, it does not deal with implementation and NFPs. Finally, Unicon provides several types of connectors (namely Pipe, FileIO, ProcedureCall, RemoteProcedureCall, DataAccess, RTScheduler and PLBundler), but it does not support different middleware.
2. *Reflecting NFPs in connectors.* Similar to our work is reflective middleware by Blair, et al. [11]. They construct connectors also from smaller parts each part implementing a dedicated connector functionality. However, as their primary aim is middleware for streaming multimedia, they address problems related to QoS such as performance monitoring, dynamic adaptation, etc. On the other hand, in our work we try to address interoperability between different platforms (and middleware).
3. *automatic connector adaptation.* Very similar to our element adaptation (performed as a final step during connector generation) is generation of stubs and skeleton in CORBA and RMI. In fact, the element adaptation (in case of stubs and skeletons) often relies on IDL- or RMI-compiler shipped with respective middleware. However, our connectors are not bound to only one specific middleware. Moreover they provide some additional services (e.g. logging, synchronization, etc.).

This paper deals mainly with automation of connector generation. Thanks to the automation, we can specify only rough requirements (i.e. communication style and NFPs) and let the generator construct a connector that satisfies them with respect to provisions of target platforms (operating system, installed libraries, middleware, etc.) This, probably the most important feature of our connector model, is to our knowledge not discussed anywhere else.

5. Summary and future work

In this paper we proposed a way to automaticaly select a connector configuration with respect to requirements specified in form of communication style and NFPs. Our approach uses backtracking to traverse a tree of possible connector configurations. A correct and suitable configuration is chosen using three kinds of rules checking that: (1) element implementations a connector will consists of are mutually compatible (composition consistency), (2) the connector will implement all request NFPs (NFP consistency), and (3) the element implementations will be able to run on a target platform (environment consistency).

The near future work comprises further elaboration on the rules, mainly the rules for environment consistency, which being currently very trivial do not allow for complex requirements.

References

[1] Sun Microsystems, Inc., *Java Remote Method Invocation Specification – Java 2 SDK, v1.4.1*, 2002.

[2] OMG formal/02-12-06, *The Common Object Request Broker Architecture: Core Specification, v3.0*, Dec 2002.

[3] IETF RFC 1050, *RPC: Remote Procedure Call Protocol Specification Version 2*, Jun 1988.

[4] T. Bures and F. Plasil, "Scalable element-based connectors," in *Proceedings of SERA 2003, San Francisco, USA*, Jul 2003.

[5] D. Balek and F. Plasil, "Software connectors and their role in component deployment," in *Proceedings of DAIS'01, Krakow, Kluwer*, Sep 2001.

[6] L. Bulej and T. Bures, "A connector model suitable for automatic generation of connectors," Tech. Rep. 2003/01, Dep. of SW Engineering, Charles University, Prague, Jan 2003.

[7] N. Medvidovic, P. Oreizy, and R. Taylor, "Reuse of off-the-shelf components in c2-style architectures," in *Proceedings of the 1997 International Conference on Software Engineering (ICSE.97), Boston, MA*, 1997.

[8] D. Garlan, R. T. Monroe, and D. Wile, "Acme: Architectural description of component-based systems," in *Foundations of Component-Based Systems* (G. T. Leavens and M. Sitaraman, eds.), pp. 47–68, Cambridge University Press, 2000.

[9] M. Shaw, R. DeLine, D. V. Klein, T. L. Ross, D. M. Young, and G. Zelesnik, "Abstractions for software architecture and tools to support them," *Software Engineering*, vol. 21, no. 4, pp. 314–335, 1995.

[10] T. Bures, "Constraint-based generation of connectors." Presented at Week of Doctoral studies, MFF UK, Jun 2003.

[11] G. Blair and et al., "A principled approach to supporting adaptation in distributed mobile environments," in *International Symposium on Software Engineering for Parallel and Distributed Systems, Limerick, Ireland*, Jun 2000.

Product Lukasiewicz Logic – An Overview

doktorand:
PETR CINTULA
(SPOLUAUTOR R. HORČÍK)

Ústav informatiky, Akademie věd Českě Republiky, Pod vodárenskou věží 2, 182 07 Prague 8

Cintula@cs.cas.cz

školitel:
PETR HÁJEK

Ústav informatiky, Akademie věd Českě Republiky, Pod vodárenskou věží 2, 182 07 Prague 8

hajek@cs.cas.cz

obor studia:
Matematické inženýrství

Abstract

Among all many-valued logics the Lukasiewicz logic plays a fundamental role. However expressive power of this logic is restricted to piecewise linear functions. In this paper we enrich the language of Lukasiewicz logic by adding a new connective which expresses multiplication. The resulting logic PL is defined and developed. We also deal with several extensions of this logic. At the end of the paper, the predicate version of PL logic is introduced and developed.

The following text is the introduction from the upcoming paper [5].

Lukasiewicz logic [7, 6] is indeed one of the most important logics from the broad family of the many-valued logics. Its corresponding algebraic structures of truth values - MV-algebras - are well-known and deeply studied. Mundici's famous result [1] established an important correspondence between MV-algebra and Abelian l-groups with strong unit. There is an obvious question if there is a logic, whose corresponding algebras of truth values are in analogous correspondence with l-rings.

There are several papers dealing with so-called product MV-algebras. Fundamental to our aims are Montagna's papers [8, 9, 11], there is also one paper by Di Nola and Dvurečenskij [3]. Product MV-algebra (PMV-algebra for short) is an MV-algebra enriched by product operation in such a way that it corresponds to f-ring with strong unit. In [8], Montagna proved subdirect representation theorem for PMV-algebras and established a correspondence between linearly ordered f-rings with strong unit and linearly ordered PMV-algebras. Later in [9], he introduced PMV$_\triangle$-algebras (PMV-algebras enriched by 0-1 projector $\triangle$) and proved categorical equivalence between PMV$_\triangle$-algebras and certain extension of f-rings (so-called δ-f-rings). Finally in [11], it was shown by Montagna and Panti that the variety of PMV$_\triangle$-algebras is generated by the standard PMV$_\triangle$-algebra (over the real unit interval).

Recently in upcoming paper [10], Montagna introduced a quasi-variety $\mathbf{PMV}^+$ containing only those PMV-algebras without non-trivial zero-divisors and showed that this quasi-variety is generated by standard

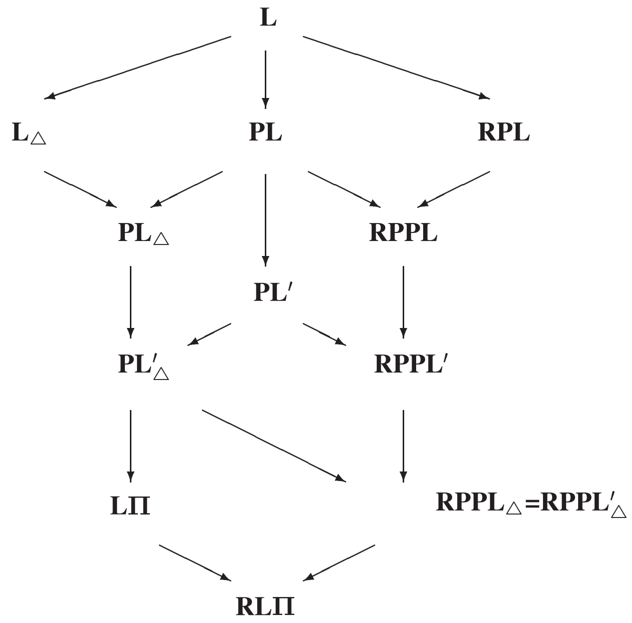


Figure 1: Relations between studied logics.

PMV-algebra (over the real unit interval).

However so far there is no logic corresponding to above mentioned algebras. The main aim of this paper is to define and develop such a logic. Our logic, which corresponds to the PMV-algebras, is called PL logic. Further we introduce PL$'$ logic corresponding to the algebras from $\mathbf{PMV}^+$. We also study extensions of PL and PL$'$ logics by Baaz's $\triangle$ so-called PL$_\triangle$ and PL$'_\triangle$. The algebras of truth values of PL$'_\triangle$ logic correspond to PMV$_\triangle$-algebras. This, together with the fact that there are also several other different algebraic structures called PMV-algebras, is the reason why we denote the algebras of truth values corresponding to PL logic PL-algebras. Analogously we denote the algebras corresponding to PL$'$, PL$_\triangle$, and PL$'_\triangle$ logics by PL$'$-algebras, PL$_\triangle$-algebras, and PL$'_\triangle$-algebras, respectively.

We use the above mentioned algebraic results to obtain completeness for all of these logics, standard completeness for PL$'$ and PL$'_\triangle$ logic. Further we show the example of PL-algebra proving that PL logic is not standard complete. Then we show a relation of our logics with well-know logic LΠ, which was defined in [4]. Roughly speaking the logic LΠ is the extension of PL$'$ by product residuum.

Furthermore we extend these logics by rational constants in the same way as the Rational Pavelka's logic (RPL) extends Lukasiewicz logic (see [13], [12], and [6, Section 3.3]). We obtain RPPL, RPPL$'$, RPPL$_\triangle$, and RPPL$'_\triangle$ logics. We prove Pavelka-style completeness of these logics and show that the logics RPPL$_\triangle$ and RPPL$'_\triangle$ coincide. Further we prove standard completeness of RPPL$_\triangle$ and show the relation of these logics with RLΠ (the extension of LΠ by rational constants; see [4]) and RPL (Rational Pavelka Logic).

Then we investigate the predicate versions of all logics mentioned above with the exception of PL$'$ (the problem is that we can prove only the completeness of this logic w.r.t. all PL$'$-algebras but we are not able to prove it w.r.t. linearly ordered PL$'$-algebras; thus we leave this problem open). We prove completeness for PL$\forall$, PL$_\triangle\forall$ and PL$'_\triangle\forall$; Pavelka style completeness for RPPL$\forall$, RPPL$_\triangle\forall$ and even standard completeness for RPPL$_\triangle\forall$. Then we deal with arithmetical complexity of the set of tautologies for these logics which gives us that the logics PL$\forall$, PL$_\triangle\forall$ and PL$'_\triangle\forall$ does not have the standard completeness property. Finally we show a relation of these logics with predicate version of LΠ, RLΠ (which were introduced in [2]) and the famous and important logic of Takeuti and Titani [14].

All investigated propositional logics lie between Lukasiewicz logic and RLΠ logic [2, 4]. Relations between them are depicted in Figure 2.

References

[1] R. Cignoli, I.M.L. D'Ottaviano, D. Mundici: *Algebraic Foundations of Many-Valued Reasoning.* Kluwer, Dordrecht, 1999.

[2] P. Cintula: *The* $L\Pi$ *and* $L\Pi\frac{1}{2}$ *propositional and predicate logics.* Fuzzy Sets and Systems 124/3:21–34, 2001.

[3] A. Di Nola, A. Dvurečenskij: *Product MV-algebras.* Multiple-Valued Logic 6:193–215, No.1-2, 2001.

[4] F. Esteva, L. Godo, F. Montagna: *The* $L\Pi$ *and* $L\Pi\frac{1}{2}$ *logics: two complete fuzzy systems joining Lukasiewicz and product logics.* Archive for Mathematical Logic 40:39–67, 2001.

[5] R. Horčík, P. Cintula: *Product Lukasiewicz logic.* To appear in Archive for Mathematical Logic.

[6] P. Hájek: *Metamathematics of Fuzzy Logic*. Kluwer, Dordrecht, 1998.

[7] J. Lukasiewicz: *O logice trojwartosciowej (On three-valued logic).* Ruch filozoficzny 5:170-171, 1920.

[8] F. Montagna: *An algebraic approach to propositional fuzzy logic.* Journal of Logic Language and Information 9:91-124, 2000.

[9] F. Montagna: *Functorial Representation Theorems for* MV_Δ *algebras with additional operators.* Journal of Algebra, 238(1):99-125, 2001.

[10] F. Montagna: *Reducts of MV-algebras with product and product residuation.* (draft)

[11] F. Montagna, G. Panti: *Adding structure to MV-algebras.* Journal of Pure and Applied Algebra 164(3):365-387, 2001.

[12] V. Novák, I. Perfilieva, J. Močkoř: *Mathematical Principles of Fuzzy Logic*. Kluwer, Norwell, 1999.

[13] J. Pavelka: *On Fuzzy Logic I,II,III.* Zeitschrift für math. Logic und Grundlagen der Math. 25:45–52,119–134,447–464, 1979.

[14] G. Takeuti, S. Titani: *Fuzzy Logic and Fuzzy Set Theory.* Archive for Mathematical Logic 32:1–32, 1992.

Radial Implicative Fuzzy Inference Systems

doktorand:
ING. DAVID COUFAL
Institute of Computer Science
david.coufal@cs.cas.cz

školitel:
DOC. ING. STANISLAV KREJČÍ, CSC.
University of Pardubice
Stanislav.Krejci@upce.cz

obor studia:
Technical cybernetics

Abstract

The paper presents the abstract of Ph.D. thesis I finished this year. Readers interested in the topic of it are referred directly to the author `david.coufal@cs.cas.cz` to obtain the thesis in the printed or electronic form.

The Ph.D. thesis deals with radial implicative fuzzy inference systems. Implicative fuzzy inference systems (I-FISs) are the fuzzy inference systems having their rules represented by genuine fuzzy implications, not by fuzzy conjunctions. Radial implicative fuzzy inference systems (radial I-FISs) are I-FISs exhibiting the radial property, i.e., having antecedents and succedents of their rules represented by a radial basis function.

Main theoretical results of the thesis are related to the building of radial I-FISs on the basis of a given t-norm and to the study of some of their properties such as coherency, redundancy and the universal approximation property.

By the theoretical results it is shown how to specify shapes of fuzzy sets employed in an I-FIS to it retains the radial property with respect to an a priori chosen Archimedean t-norm (used for the specification of antecedents of I-FIS's rules). The radial property enables to specify a computationally very easy algorithm for testing the coherency of a system, i.e., for testing the non-emptiness of output of a system with respect to an arbitrary input. The radiality also enables to specify an algorithm for detection of redundant rules of a system which are unimportant for its computation. Finally, it is shown that radial I-FISs are universal approximators, i.e., that they are able to approximate any continuous function to a given degree. Moreover, it is shown that they are even able to approximate two such functions employing the same number of parameters.

The thesis are accompanied by results related to a practical application of the theoretical considerations. There are presented algorithms for structure and parameter learning of radial I-FISs based on well known algorithms of fuzzy clustering and non-linear least squares optimization. These algorithms are implemented in the MATLAB language and its application is demonstrated on several examples in experimental part.

Strukturované neuronové sítě

doktorand:
ING. MAREK HLAVÁČEK
ING. ROMAN KALOUS

Katedra matematiky, FJFI ČVUT, Praha

hlavacek@km1.fjfi.cvut.cz
kalous@kmlinux.fjfi.cvut.cz

školitel:
ING. FRANTIŠEK HAKL, CSC.

ÚI, AV ČR, Praha

hakl@cs.cas.cz

obor studia:
Neuronové sítě
Stochastické optimalizace

Abstrakt

Článek shrnuje základní poznatky o strukturovaných neuronových sítích s přepínacími jednotkami, které představují zobecnění původního modelu lineárního řetězce. Jeho hlavním cílem je představit způsob reprezentace obecné architektury strukturované neuronové sítě. Nalezení vhodné reprezentace je totiž nezbytné pro následnou optimalizaci architektury neuronových sítí pomocí genetického algoritmu.

1. Úvod

Naším cílem je nalézt třídu neuronových sítí, v rámci níž by bylo možné optimalizovat architekturu sítí tak, aby co nejlépe řešila konkrétní problém separace množin. S ohledem na charakter úlohy jsme jako optimalizační nástroj zvolili genetický algoritmus. Jeho použití je podmíněno možností rychlého naučení a ohodnocení kvality neuronových sítí a existencí dostatečně obecné reprezentace architektury neuronové sítě.

Zejména kvůli rychlosti učení jsme vyšli z modelu *lineárního řetězce s přepínacími jednotkami* poprvé popsaného v [BSK95] využívajícího k optimalizaci svých parametrů lineární regresi, která je podstatně rychlejší než běžně používané gradientní metody. Vzhledem k velkému významu tohoto modelu je pojem lineárního řetězce formálně zaveden, a to včetně rozšíření o takzvané *korekční jendotky,* v části 2.1. Naše experimenty ukázaly, že lineární řetězce jsou sice relativně silné nástroje pro řešení různých problémů, ale bohužel v kontextu s genetickou optimalizací nepředstavují dostatečně širokou třídu sítí.

Bylo tedy nutné další zobecnění. Tím jsou takzvané *strukturované neuronové sítě* (SNS). Základní myšlenka SNS spočívá v propojení více lineárních řetězců a jiných elementárních neuronových sítí. Výsledná neuronová síť je tedy strukturována do bloků, které sami o sobě představují neuronové sítě. V průběhu genetické optimalizace je pak možné optimalizovat zvlášť propojení jednotlivých bloků a jejich vlastnosti. Aby však bylo možné efektivně využít strukturované topologie bylo nutné zavést její vhodnou reprezentaci. Jako nevyhovující se ukázala adjacenční matice. Ta totiž neumožňuje efektivně manipulovat se substrukturami sítě a při generovaní acyklických orientovaných grafů je navíc nutné kontrolovat, jestli daná matice skutečně reprezentuje acyklický graf. Rozhodli jsme se proto pro takzvané *celulární kódování* (viz [Gruau94]), které jsme dále rozšířili o další parametry a implementovali pomocí zobecněných *readových kódů* (viz [Read72]). Popisem SNS a jejich reprezentací se podrobně zabývá část 3.1. Některé příklady netriviálních architektur, jejich vlastnosti a vazby na Readovy kódy jsou pak prezentovány v části 4.

2. Lineární řetězec s přepínacími a korekčními jednotkami

K tomu, abychom kompletně popsali libovolnou neuronovou síť, musíme popsat neurony, z kterých se síť skládá, jejich vzájemné propojení, neboli *topologii sítě*, a neposlední řadě její *dynamiku*, která definuje akci neuronu na libovolný vstup a způsob učení sítě. V následujícím textu budeme rovněž používat pojem *architektura neuronové sítě*, kterým označujeme souhrnný popis vlastností jednotlivých neuronů a topologie sítě.

2.1. Neuron s přepínací jednotkou

V souladu s úvodní poznámkou začneme nejprve popisem jednotlivých neuronů, přičemž vyjdeme z následující definice obecného neuronu.

Definice 1 *Nechť jsou dána čísla $n, m, l \in \mathbb{N}$, která udávají dimenzi vstupu, výstupu a parametru neuronu. Dále nechť je dána funkce $F : \mathbb{R}^l \to \mathbf{T}(\mathbb{R}^n, \mathbb{R}^m)$, kterou nazveme typ neuronu a parametr $p \in Dom\,(F)$, jehož obraz $F(p)$ definuje přechodovou funkci neuronu. Uspořádanou dvojici $[F, p]$ nazveme neuron. Akce neuronu na libovolný vstup $x \in \mathbb{R}^n$ je dána předpisem $\mathbb{R}^m \ni y := [F\,(p)]\,(x)$.*

Každý neuron tedy představuje jistou parametrickou funkci zobrazující vstupy neuronu na jeho výstupy. Akce neuronu je dána jednak jeho typem a jednak konkrétní hodnotou parametru. Při popisu sítě však stačí definovat typ neuronu, neboť ten definuje chování neuronu pro libovolnou hodnotu parametru. Na základě s předchozí definice již můžeme formálně zavést neuron s přepínací jednotkou.

Definice 2 *Nechť je dáno:*

- *čísla $n, m, l, s, k \in \mathbb{N}$, která udávají vstupní a výstupní dimenzi neuronu, dimenzi parametrů výpočetních jednotek, dimenzi parametrů jednotlivých klastrů a počet klastrů (respektive výpočetních jednotek),*
- *funkce $S : (\mathbb{R}^s)^k \to \left(2^{\mathbb{R}^n}\right)^k$ taková, že pro všechna $p_c \in Dom\,(S)$ je $\left\{S^1\,(p_c)\,, ..., S^k\,(p_c)\right\}$ disjunktní pokrytí $\mathbb{R}^n$, která se nazývá typ přepínací jednotky,*
- *funkce $\bar{F} : \mathbb{R}^l \to \mathbf{T}(\mathbb{R}^n, \mathbb{R}^m)$, kterou nazveme typ výpočetní jednotky.*

Dále nechť je pro všechna $p \in P := Dom(S) \times \left(Dom(\bar{F})\right)^k$, $p = (p_c, p_1, \ldots, p_k)$ definována funkce $F : P \to \mathbf{T}(\mathbb{R}^n, \mathbb{R}^m)$ předpisem:

$$[F\,(p)]\,(x) = \sum_{i=1}^{k} \chi_{S^i(p_c)}\,(x)\,\left[\bar{F}\,(p_i)\right]\,(x)$$

a nechť $\widetilde{p} \in P$. Potom uspořádanou dvojici $[F, \widetilde{p}]$ nazveme neuronem s přepínací jednotkou a k klastry.

Každý neuron s přepínací jednotkou se tedy skládá z jedné přepínací jednotky a jedné nebo více výpočetních jednotek (viz obr. 1). Každá výpočetní jednotka představuje "klasický" neuron, zatímco pojmem přepínací jednotka označujeme “zařízení”, které distribuuje vstupy mezi jednotlivé výpočetní jednotky. Přepínací jednotka totiž definuje disjunktní rozklad vstupního prostoru a výraz $\chi_{S^i(p_c)}(x)$, který vystupuje v předcházející definici 2, zaručuje, že se na zpracování daného vstupu x podílí pouze jediná, i-tá výpočetní jednotka neuronu s přepínací jednotkou. Tímto se neuronové sítě s přepínacími jednotkami podobají rozhodovacím stromům. Z definice rovněž vyplývá, že typ neuronu s přepínací jednotkou je popsán jednak typem přepínací jednotky (S) a jednak typem a počtem výpočetních jednotek ($\bar{F}$, k), přičemž v námi uvažovaných modelech předpokládáme, že všechny výpočetní jednotky jednoho neuronu mají stejný typ (liší se pouze jejich parametry, jinak by bylo nutné uvažovat místo jediné funkce $\bar{F}$ sytém k funkcí $\bar{F}_1, ..., \bar{F}_k$).

Typ neuronu s přepínací jednotkou, který je použit v lineárním řetězci můžeme popsat následovně. Předpokládejme, že vstupem neuronu může být libovolný vektor z $\mathbb{R}^n$. Následující vztah definuje typ přepínací jednotky $S_{|.|}$ s k klastry:

$$S^i_{|.|}(p_c) := \left\{ x \in \mathbb{R}^n \;\middle|\; i = \min_{q=1,\ldots,k} \left\{ \left| \Big(\sum_{j=1}^n x^j\Big) - p_c^q \right| = \min_{l=1,\ldots,k} \left| \Big(\sum_{j=1}^n x^j\Big) - p_c^l \right| \right\} \right\}, \forall i = 1, \ldots, k, \tag{1}$$

kde $p_c \in \mathbb{R}^k$. Typ výpočetní jednotky pak pro každé $p \in \mathbb{R}^{n+1}$ definuje rovnost

$$\left[\bar{F}_L(p)\right](x) := \left(p^1 x^1, p^2 x^2, ..., p^n x^n, p^{n+1}\right), \; \forall x \in \mathbb{R}^n. \tag{2}$$

Z geometrického hlediska si lze funkci výše definovaného typu neuronu $F_L(.)$ představit tak, že přepínací jednotka nejprve rozdělí vstupní prostor na několik pásů (resp. na několik pásů a dvě poloroviny, v případě dimenze 2). Následně vektor parametrů příslušné výpočetní jednotky definuje pro každý takový pás nadrovinu s rovnicí $W \equiv p^{n+1} + \sum_{i=1}^n x^i p^i = 0$, která tento pás dále dělí na “kladnou” a “zápornou” část. Výstup neuronu je pak kladný právě když vstup leží v kladné části odpovídajícího klastru.

Pokud je úkolem sítě separovat dvě množiny, jsou během učení sítě nastavovány parametry neuronů tak, aby do “kladných” částí klastrů mapovaly prvky z jedné množiny a do “záporných” prvky patřící do druhé množiny. V každém okamžiku lze tedy na základě znaménka součtu složek výstupu rozhodnout, do které z množin by daný neuron zařadil daný vstup. Tento fakt je velice důležitý v kontextu s korekčními jednotkami.

Úkolem korekčních jednotek je odstranění nežádoucího efektu plynoucího z přítomnosti přepínacích jednotek v neuronové síti. Představme si například, že chceme separovat dvě množiny, nazvěme je signál a pozadí. Z definice 2 vyplývá že, neuron s přepínací jednotkou lze považovat za neuron s po částech definovanou přechodovou funkcí (v případě lineárního řetězce se jedná o po častech lineární funkci). Takto definovaná přechodová funkce však nemusí být prostá a proto se může velice snadno stát, že jistá oblast signálu odpovídající některému z klastrů je mapována do stejné oblasti, jako oblast pozadí příslušející jinému klastru. Toto “překrytí” signálu a pozadí brání další separaci překrývajících se oblastí. Cílem korekčních jednotek je uchovat informaci neuronu (resp. části sítě) o separaci signálu a pozadí a zároveň transformovat data tak, aby je bylo možné dále separovat. Budeme-li předpokládat, že původní data jsou separovatelná a jsou jsou částí krychle $(-1, 1)^n$, stačí například posunout vstupy, které neuron označí jako signál do kladného oktantu a ostatní (pozadí) naopak do záporného. Přesněji zavádí pojem korekční jednotky následující definice.

Definice 3 *Nechť $n = n_1 + n_2$ je dimenze vstupu korekční jednotky a nechť $n_1, n_2 > 0$. Potom pro libovolné $x \in \mathbb{R}^{n_1} \times \mathbb{R}^{n_2}$ a $p \in \mathbb{R}^{n_2}$ je akce korekční jednotky dána předpisem*

$$\mathbb{R}^{n_2} \ni [F_C(p)](x) := \left[sgn\left(\sum_{i=1}^{n_1} x^i \right) \right] (1, ..., 1) + \left(\frac{x^{n_1+1}}{p^1}, ..., \frac{x^{n_1+n_2}}{p^{n_2}} \right).$$

K této definici poznamenejme ještě, že posledních n_2 vstupů představuje zpravidla (v lineárním řetězci vždy) vstupy sítě, zatímco prvních n_1 vstupů představuje výstup předcházejícího neuronu. Podrobněji se vzájemným propojením jednotlivých neuronů budeme zabývat v následující podsekci, která popisuje topologii lineárního řetězce a jeho dynamiku.

2.2. Topologie a dynamika lineárního řetězce

Topologii lineárního řetězce s přepínacími jednotkami (viz [BSK95]) lze popsat acyklickým orientovaným grafem (AOG), který je cestou (tj. každý vrchol s vyjímkou prvního a posledního má právě jednoho potomka a právě jednoho předka). Každý vrchol pak představuje jeden neuron s přepínací jednotkou a libovolným počtem klastrů.

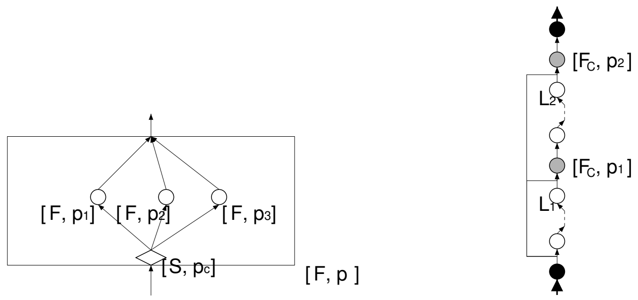
O něco komplikovanější je topologie lineárního řetězce s přepínacími a korekčními jednotkami. V podstatě se jedná o několik lineárních řetězců s přepínacími jednotkami, které jsou mezi sebou propojeny pomocí korekčních jednotek a to tak, že poslední neuron prvního řetězce je předek první korekční jednotky a první neuron druhého řetězce je potomkem této korekční jednotky. Druhá korekční jednotka pak spojuje stejným způsobem druhý řetězec s třetím a tak dále. Navíc do každé přepínací jednotky vede kromě hrany z předcházejícího řetězce také hrana přímo ze vstupu sítě. Celý řetězec pak může být ukončen korekční jednotkou nebo řetězcem s přepínacími jednotkami (viz obr. 1).

Nyní nám již zbývá popsat pouze dynamiku lineárního řetězce. Vzhledem k tomu, že se jedná o neuronovou síť s acyklickou topologií je přirozené, že je signál zpracováván sekvenčně směrem od vstupního neuronu (odpovídá vrcholu, který nemá žádné předky) přes jeho potomky až po výstupní neuron (nemá žádné potomky).

Učení lineárního řetězce probíhá analogicky jako zpracování konkrétního vstupu. Rozdíl spočívá pouze v tom, že do daného neuronu je v jeden časový okamžik přivedena celá tréninková množina (respektive její obraz získaný průchodem sítí), na základě níž jsou definitivně nastaveny všechny parametry neuronu (jak přepínací jednotky, tak všech výpočetních jednotek). Následně jsou všechny vstupy transformovány již naučeným neuronem a předány dále do sítě. To znamená, že po jednom průchodu učící množiny sítí je tato síť naučena. K učení přepínacích jednotek se v případě lineárního řetězce využívá Janceyův klastrovací algoritmus zatímco k nastavení parametrů výpočetních jednotek je využívána metoda lineární regrese. Zbývá nám tedy popsat způsob učení korekčních jednotek. Ale protože můžeme bez újmy na obecnosti předpokládat, že vstupy sítě jsou normovány tj. leží v krychli $(-1, 1)^n$ není učení korekčních jednotek nutné (parametr lze volit jako $(1, \ldots, 1) \in \mathbb{R}^n$).

Na závěr této sekce poznamenejme, že zde prezentovaný popis dynamiky lineárního řetězce je společný pro všechny námi uvažované sítě. Lišit se může pouze algoritmus pro nastavení parametrů jednotlivých neuronů (tj. místo lineární regrese může být například použita libovolná nelineární optimalizační metoda).

V následujících sekcích budou popsány obecnější dopředné neuronové sítě s přepínacími jednotkami a způsob jejich reprezentce.



Obrázek 1. Neuron s přepínací jednotkou a třemi výpočetními jednotkami (vlevo) a topologie lineárního řetězce s přepínacími a korekčními jednotkami (vpravo).

3. Reprezentace

Topologie lineárního řetězce je jako graf dána velmi konkrétně a jediným parametrem při konstrukci je jeho délka. Naší snahou bylo nalézt komplikovanější topologie a bylo proto třeba stanovit, jaké grafy pokládáme za topologie. Množina námi požadovaných topologií jsou acyklické orientované grafy a to pouze takové grafy, jejichž všechny uzly mají alespoň jednoho rodiče a alespoň jednoho potomka (vyjma těch, které považujeme za vstup a výstup sítě).

Definice 4 *Acyklickou topologií rozumíme acyklický orientovaný graf* $\mathcal{G} = (\{IN, OUT\} \cup V, E)$, $V \neq \emptyset$, *takový, že existuje orientovaná cesta z* IN *do* OUT *a pro libovolný vrchol* $j \in V$ *existuje orientovaná cesta z* IN *do* j *a cesta z* j *do* OUT.

Protože je možné a v některých situacích výhodné pracovat namísto s jednotlivými neurony se skupinami neuronů – *bloky*, které jsou dány indukovanými podgrafy v dané acyklické topologii, nazýváme neuronové sítě s acyklickými topologiemi *strukturované neuronové sítě* (SNS). Acyklická topologie, společně s dalšími přídavnými parametry jednotlivých neuronů (nejčastěji počet klastrů příslušné přepínací jednotky, specifikace výběru proměnných definované v kapitole 4), představuje *acyklickou architekturu* SNS.

Při hledání reprezentace architektury SNS a její implementace jsme záměrně nevyužili nativní reprezentace grafu – adjacenční matice, protože neumožňuje efektivní náhodné generování acyklických topologií, modifikace substruktur, apod.. Navíc, jedním z cílů vývoje SNS byla optimalizace procesů hledání submodelů a klasifikace kvality dat pomocí genetického algoritmu a v tomto ohledu se adjacenční matice ukázaly rovněž jako nevyhovující. Zavedení operátorů genetického algoritmu křížení, mutace, které jsou většinou definovány jako záměny/změny podčástí dvou reprezentací, je v případě adjacenčních matic velmi komplikované, protože změnou adjacenční matice acyklické topologie nemusíme obecně znovu získat adjacenční matici odpovídající acyklické topologii.

3.1. Instrukční kódy

Druh reprezentace acyklické topologie, který splňuje předchozí nároky, je dán na rozdíl od adjacenční matice nikoli jako souhrn popisných informací o acyklické topologii, ale jako informace o způsobu její konstrukce. Jedná se o *celulární kódování* (CK) zavedené F. Gruauem v práci [Gruau94].

CK je vlastně posloupnost konstrukčních příkazů, která je uspořádána tak, aby bylo zřejmé, v jakém pořadí při sestavování topologie postupovat a jakých substruktur se konstrukční příkazy týkají. Konstrukční příkazy nazveme *instrukcemi*. Pořadí instrukcí je dáno pozicí v *pěstovaném stromu*[1], resp. pořadím uzlů, v nichž jsou instrukce uloženy, při procházení tohoto stromu. Protože je strom zvolen jako pěstovaný, je pořadí instrukcí dáno jednoznačně. Instrukce je možné kódovat například vhodným označením vrcholů a tak je možné ztotožnit CK s pěstovaným stromem, který budeme označovat $\mathcal{T}$.

Konstrukce acyklické topologie dle CK je založena na jednoduchém rekurzivním schématu, jehož vstupem je v každém kroku konstruovaný graf $\mathcal{G} = (V, E)$, příslušné CK dané stromem s instrukcemi $\mathcal{T}$ a určení aktuálně zpracovávaného uzlu $j \in V(\mathcal{G})$. Postup konstrukce je potom dán následujícím algoritmem:

1. je-li kořen $\mathcal{T}$ list, pak již uzel j nepodléhá dalším konstrukcím
2. není-li kořen $\mathcal{T}$ list a instrukce je 'P', pak je uzel j zdvojen 'vedle sebe':
   (a) $V(\mathcal{G}) = V(\mathcal{G}) \cup \{k = \min\{n \in \mathbb{N} \mid n > j \in V(\mathcal{G})\}\}$
   (b) $E(\mathcal{G}) = E(\mathcal{G}) \cup \{(l, k) \mid l \in \mathrm{A}_{\mathcal{G}}(j)\} \cup \{(k, l) \mid l \in \mathrm{D}_{\mathcal{G}}(j)\}$
   (c) rekurzivní aplikace algoritmu na graf $\mathcal{G}$, CK dané levým podstromem, který vznikne odebráním kořene z $\mathcal{T}$, uzel j a
   rekurzivní aplikace algoritmu na graf $\mathcal{G}$, CK dané pravým podstromem, který vznikne odebráním kořene z $\mathcal{T}$, uzel k
3. není-li kořen $\mathcal{T}$ list a instrukce je 'S', pak je uzel j zdvojen 'za sebou'

[1]Pěstovaný strom je trojice $t = (T_t, r_t, \nu_t)$, kde T_t je strom, $r_t \in V(T_t)$ je vrchol označený jako kořen a ν_t je pevné zakreslení stromu T_t v rovině. Ekvivalence pěstovaných stromů je tedy silnější než ekvivalence stromů, což později umožní existenci jednoznačného přiřazení acyklické topologie a CK.

(a) $V(\mathcal{T}) = V(\mathcal{G}) \cup \{k = \min\{n \in \mathbb{N} \mid n > j \in V(\mathcal{G})\}\}$
(b) $E(\mathcal{T}) = (E(\mathcal{G}) \cup (j,k) \cup \{(k,l) \mid l \in \mathrm{D}_{\mathcal{G}}(j)\}) \setminus \{(l,j) \mid l \in \mathrm{A}_{\mathcal{G}}(j)\}$
(c) rekurzivní aplikace algoritmu na graf $\mathcal{G}$, CK dané levým podstromem, který vznikne odebráním kořene z $\mathcal{T}$, uzel j a
rekurzivní aplikace algoritmu na graf $\mathcal{G}$, CK dané pravým podstromem, který vznikne odebráním kořene z $\mathcal{T}$, uzel k

Algoritmus 2. Algoritmus konstrukce acyklické topologie dle CK. $\mathrm{A}_{\mathcal{G}}(j)$, resp. $\mathrm{D}_{\mathcal{G}}(j)$ reprezentují množinu rodičů, resp. potomků uzlu j v grafu $\mathcal{G}$.

Tento algoritmus provedeme pro iniciační graf $\mathcal{G}_{INIT} = (\{\mathrm{IN}, 1, \mathrm{OUT}\}, \{(\mathrm{IN}, 1), (1, \mathrm{OUT})\})$, nějaké CK a vrchol 1. Výsledkem algoritmu je acyklická topologie definující nějakou SNS.

Z algoritmu vidět, že $\mathcal{T}$ je binární pěstovaný strom, protože počet podstromů použitých v každém kroku je dán počtem uzlů, které dle instrukcí vzniknou. Protože jsou použity instrukce pro zdvojení uzlů a jiné instrukce nejsou implementovány (například smazání daného uzlu, rekurzivní volání původního CK pro daný uzel), budou v dalším uvažovány pouze binární pěstované stromy CK.

Takto definované CK není jednoznačně svázáno s výsledným grafem, protože není využita jednoznačná vazba listů CK a uzlů výsledného grafu. Listy nebudou obsahovat instrukce, ale parametry neuronů výsledné sítě; tím se stává CK reprezentací *architektury* SNS. Jednoduchý příklad konstrukce architektury SNS dle CK je na obrázku 3.

Pro implementaci mechanismu CK v programu je výhodnější pěstované stromy i s příslušnými přídavnými parametry v listech uchovávat ve tvaru posloupnosti, která umožňuje snadné zacházení s těmito strukturami. Je použit postup rozšířených *readových kódů* (RK), které zavedl R. C. Read pro obecné pěstované stromy (viz [Read72]). RK pěstovaného stromu t, který označíme $RK(t)$, je dán jako posloupnost hodnot $0, \ldots, \max_{j \in V(T_t)}\{\mathrm{D}_{T_t}(j)\}$, kde $\mathrm{D}_{T_t}(j)$ je počet potomků uzlu j stromu T_t. Tato posloupnost se konstruuje rekurzivně pomocí pravidla

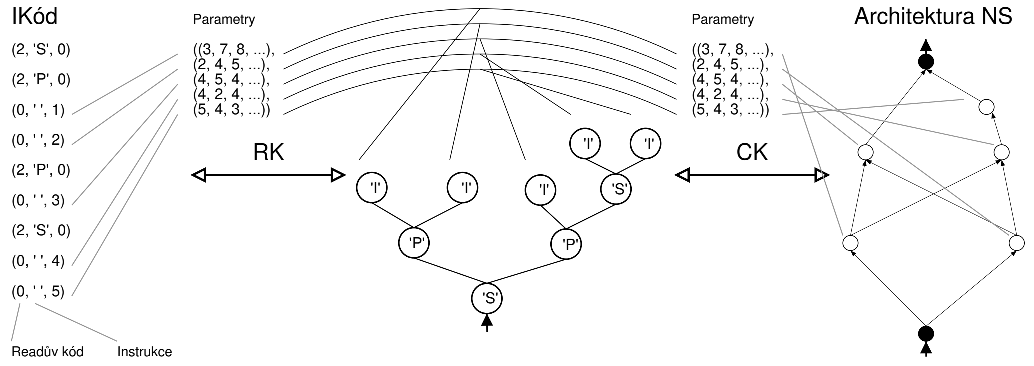
$$RK(t) = \left\{ D_{T_t}(r_t), RK(t_1), \ldots, RK(t_{D_{T_t}(r_t)}) \right\}. \tag{3}$$

Pořadí $RK(t_j), j = 1, \ldots, D_t(r_t)$, potomků kořene stromu t je jednoznačně dáno zakreslením pěstovaného stromu do roviny. RK listu (stromy isomorfní s $t = \{0, \emptyset\}$) je dán vztahem $RK(t) = 0$. Například RK úplného binárního stromu hloubky 2 je dán $\{2,\ 2,\ 0,\ 0,\ 2,\ 0,\ 0\}$.

Počet prvků posloupnosti $RK(t)$ je roven počtu uzlů stromu t. Pro množinu pěstovaných stromů je zobrazení $RK(.)$ bijekcí množiny pěstovaných stromů a tzv. *grafových posloupností*[2]. Náhodná grafová posloupnost potom odpovídá náhodnému pěstovanému stromu, navíc je díky shodnosti počtu prvků posloupnosti s počtem uzlů stromu možné kontrolovat i počet uzlů ve vzniklém grafu.

Protože se jednoznačnost přiřazení grafových posloupností a pěstovaných stromů zachová i ve speciální třídě binárních pěstovaných stromů, je možné kromě skalárních hodnot RK zachovávat i přídavné parametry. Konkrétně je potřeba zachovat pro vnitřní uzly informaci o instrukci při sestavování výsledné acyklické architektury a pro listy informaci o hodnotách parametrů neuronů. Pro CK $\mathcal{T}$ tedy zavedeme posloupnost trojic $(RK(\mathcal{T})_j,\ Inst(\mathcal{T})_j,\ Par(\mathcal{T})_j)_{j=1}^{q},\ q = |V(T_{\mathcal{T}})|$, z nichž první hodnota udává hodnotu Readova kódu, druhá instrukci daného uzlu a třetí parametry neuronů. Instrukce existují pouze pro vnitřní uzly stromu, proto $RK(\mathcal{T})_j = 0 \implies Inst(\mathcal{T})_j = 0,\ j = 1, \ldots, |V(T_{\mathcal{T}})|$. Neprázdné instrukce je možné reprezentovat hodnotami z $\mathbb{N}$ nebo nějakou abecedou znaků. Naopak parametry existují pouze pro listy, proto $RK(\mathcal{T})_j > 0 \implies Par(\mathcal{T})_j = 0,\ j = 1, \ldots, |V(T_{\mathcal{T}})|$. Neprázdné hodnoty parametrů jsou většinou vícerozměrné a reálné, a proto $\mathrm{Par}(\mathcal{T})_j \in \mathbb{N}$ udává odkaz na pozici v poli polí reálných parametrů všech výpočetních jednotek. Popsanou posloupnost trojic nazveme *instrukční kód*, zkráceně IKód.

[2] RK splňují tzv. *level property* – pro lib. pěstovaný strom t platí, že součet prvních d hodnot $RK(t)$ je větší nebo roven d a rovnost nastává pro $d = |V(T_t)|$. RK je v tomto smyslu grafovou posloupností CK.



Obrázek 3. Schéma zobrazení RK a CK. IKód a CK jsou ve vzájemně jednoznačném vztahu.

Počet uzlů ve výsledné acyklické architektuře SNS je v jednoznačném vztahu s IKódem, protože počet uzlů výsledné architektury je pevně dán počtem listů v IKódu. Pro počet uzlů v grafu acyklické architektury a počet prvků IKódu IC platí vztah

$$|V(\mathcal{G}_{SNS})| = \frac{|IC| + 1}{2} \tag{4}$$

Pomocí tohoto vztahu můžeme účinně kontrolovat počty neuronů v konstruované architektuře SNS.

4. Implementované netriviální acyklické architektury

Pomocí IKódů je možné pracovat s rozsáhlou třídou architektur, vhodnou modifikací podmínek kladených na RK část IKódu (kromě level property) lze dosáhnout i speciálních topologií. Z prozatím testovaných architektur sestavovaných dle IKódů uvedeme *náhodnou*, *vrstevnatou* architekturu a architekturu sítí, které jsou užívány k separaci a klasifikaci kvality dat získaných simulacemi připravovaného detektoru ATLAS, nazývanou architektura *s výběrem proměnných*.

Kromě typu neuronu F_L definovaného v části 2.1 vztahy 1 a 2 je v architekturách používán typ neuronu *s výběrem proměnných*. Formálně tento typ můžeme pro neuron s vstupní dimenzí $n \in \mathbb{N}$ definovat pomocí pevného parametru $p \in \{0,1\}^n$ jako zobrazení $F^p_{VS} : \mathbb{R}^n \longrightarrow \mathbb{R}^{\sum_{j=1}^n p_j}$, které je pro každé $x \in \mathbb{R}^n$ dáno vztahem

$$\left([F^p_{VS}](x)\right)_k = x_{\tilde{l}}, \;\; \tilde{l} = \min_{l \in \{1,\ldots,n\}} \left\{ \sum_{j=1}^{l} p_j = k \right\}, \;\; k = 1, \ldots, \sum_{j=1}^{n} p_j. \tag{5}$$

Parametr p tedy svými nenulovými složkami vytváří masku, která určuje výstupní proměnné neuronu, přičemž tyto proměnné mají stejnou hodnotu jako příslušné vstupy; tento parametr je označen jako pevný, protože je stanoven před konstrukcí sítě a není dále modifikován (například procesem učení).

4.1. Náhodné architektury

Náhodné architektury jsou získávány konstrukcí dle IKódů, které jsou náhodně generovány pomocí *level property*. Jsou použity běžné neurony bez korekčních jednotek a z parametrů jsou nastavovány pouze počty klastrů v přepínacích jednotkách.

Na začátku generování je pevně zvolen počet neuronů ve výsledné architektuře (např. náhodným výběrem čísla $\mathrm{N} \in \{\mathrm{N}_{min}, \ldots, \mathrm{N}_{max}\}$), pak je užitím level property a vztahu 4 postupně sestavován IKód IC délky $2\mathrm{N} - 1$. Tento postup je realizován od prvního prvku IC; část RK_{IC} je generována náhodně tak, aby tvořený RK v každém okamžiku splňoval level property. Jestliže je aktuálně nagenerovaná pozice RK v IC nenulová, tj. je rovna 2, odpovídá tato část vnitřnímu uzlu CK a tedy instrukci. Hodnota instrukce je (rovnoměrně) náhodně zvolena ('S' nebo 'P') a parametry jsou prázdné. Jestliže je aktuálně nagenerovaná pozice RK v IC nulová, odpovídá tato část listu CK a tedy již konkrétnímu neuronu. Instrukce je ponechána prázdná a je zvolen počet klastrů v neuronu. Nakonec je dle IC sestavena architektura SNS, která je učena a ohodnocena.

Tyto náhodně generované architektury je možné optimalizovat pomocí genetického algoritmu (GA); reprodukční schémata pak spočívají v modifikacích nebo záměnách podstromů CK včetně instrukcí a parametrů.

4.2. Vrstevnaté architektury

Zvolme počet vrstev $\mathrm{K} \in \mathbb{N}$ a pro každou vrstvu počet neuronů $\mathrm{N}_1, \ldots, \mathrm{N}_\mathrm{K} \in \mathbb{N}$. Graf vrstevnaté architektury je dán množinou uzlů $\{\mathrm{IN}, 1, \ldots, \mathrm{N}_1, \mathrm{N}_1 + 1, \ldots, \sum_{j=1}^{\mathrm{K}} \mathrm{N}_j, \mathrm{OUT}\}$ a následujícími podmínkami pro hrany:

- mezi uzly $\{\mathrm{IN}\}$ a $\{1, \ldots, \mathrm{N}_1\}$ je úplný bipartitní graf,
- pro všechna $l = 1, \ldots, \mathrm{K}$ je mezi uzly $\{\sum_{j=1}^{l-2} \mathrm{N}_j + 1, \sum_{j=1}^{l-1} \mathrm{N}_j\}$ a $\{\sum_{j=1}^{l-1} \mathrm{N}_j + 1, \sum_{j=1}^{l} \mathrm{N}_j\}$ úplný bipartitní graf,
- mezi uzly $\{\sum_{j=1}^{\mathrm{K}} \mathrm{N}_j + 1, \mathrm{OUT}\}$ a $\{1, \ldots, \mathrm{N}_1\}$ je úplný bipartitní graf.

Typy mohou být běžné neurony bez korekčních jednotek, v první vrstvě je možné použít neurony s výběrem proměnných.

V tomto případě již neplatí, že libovolný IKód nutně reprezentuje vrstevnatou síť. Je třeba klást na IKód dodatečné podmínky. Platí, že pro $\mathrm{K} \in \mathbb{N}$ a $\mathrm{N}_1, \ldots, \mathrm{N}_\mathrm{K} \in \mathbb{N}$ reprezentuje IKód

$$\begin{gathered} (2, `S', 0)(2, `P', 0)(0, `\ ', 1) \ldots (2, `P', 0)(0, `\ ', \mathrm{N}_1 - 1)(0, `\ ', \mathrm{N}_1) \\ (2, `S', 0)(2, `P', 0)(0, `\ ', \mathrm{N}_1 + 1) \ldots (2, `P', 0)(0, `\ ', \mathrm{N}_1 + \mathrm{N}_2 - 1)(0, `\ ', \mathrm{N}_1 + \mathrm{N}_2) \\ \vdots \\ (2, `P', 0)(0, `\ ', (\textstyle\sum_{j=1}^{\mathrm{K}-1} \mathrm{N}_j) + 1) \ldots (2, `P', 0)(0, `\ ', (\sum_{j=1}^{\mathrm{K}} \mathrm{N}_j) - 1)(0, `\ ', (\sum_{j=1}^{\mathrm{K}} \mathrm{N}_j)) \end{gathered} \tag{6}$$

architekturu vrstevnaté sítě s K vrstvami a $\mathrm{N}_j, j = 1, \ldots, \mathrm{K}$, neurony v j-té vrstvě. Generování IKódu pro vrstevnatou síť probíhá tedy pro náhodně zvolené počty vrstev a neuronů v jednotlivých vrstvách tak, aby RK splňoval level property a celý IKód byl navíc tvaru uvedeného v (6).

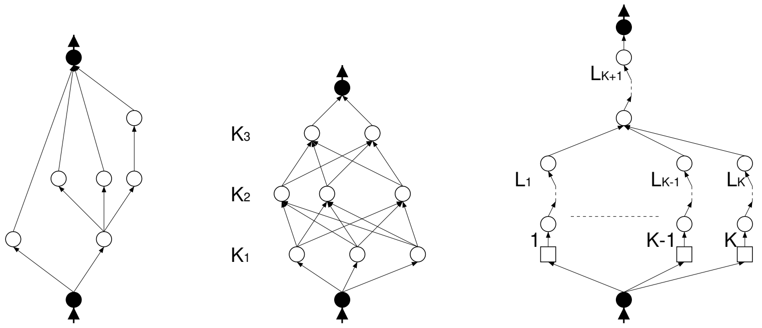
4.3. Architektury s výběrem proměnných

Architektura s výběrem proměnných je používána k testování a analýze dat problému detekce Higgsova bosonu. Jsou zde spojeny dva přístupy; jednak je jako substruktura použit lineární řetězec, který zpracovává robustně i takto komplikovaná data, a dále jsou zařazeny neurony s výběrem proměnných, které umožňují hledání submodelů a jejich kombinací.

Graf architektury je ze vstupního uzlu IN rozvětven do $\mathrm{K} \in \mathbb{N}$ neuronů s výběrem proměnných. Dle dalších částí grafu architektury rozdělujeme na

- **jednodušší typ:** za každým z neuronů s výběrem proměnných následují lineární řetězce stanovených délek $L_1, \ldots, L_K \in \mathbb{N}$. Jejich výstupy jsou pak spojeny do posledního lineárního řetězce délky $L_{K+1} \in \mathbb{N}$ a jeho výstup vede do výstupního uzlu OUT.

- **složitější typ:** za každým s neuronů s výběrem proměnných následují náhodně generované podsítě s maximálními počty neuronů $S_1, \ldots, S_K \in \mathbb{N}$. Výstupy podsítí jsou spojeny do poslední podsítě s maximálním počtem neuronů S_{K+1}, jejíž výstup vede do výstupního uzlu OUT.



Obrázek 4. Ukázky jednotlivých architektur – náhodné, vrstevnaté a s výběrem proměnných (jednodušší typ).

5. Závěr

Studiem základní architektury lineárního řetězce jsme jednoznačně prokázali, že možnosti neuronových sítí s přepínacími jednotkami nejsou zdaleka vyčerpány a že se nabízí celá řada možností pro jejich zobecnění.

O potenciálu, který nabízí neurony s přepínacími jednotkami, nás přesvědčily i naše experimenty. Ty ukázaly, že již lineární řetězce dosahují výsledků srovnatelných se standardními modely například při jejich aplikaci na separaci dat nebo na predikci sezónních časových řad.

Zobecnění lineárních řetězců, které představují dopředné strukturované neuronové sítě, pak posunulo možnosti využití neuronových sítí s přepínacími jednotkami opět o něco dále. Jejich význam navíc umocňuje prezentovaná reprezentace těchto sítí pomocí Readových kódů a následná možnost genetické optimalizace architektury sítí. Dohromady totiž genetické algoritmy a neuronové sítě představují komplexní nástroj pro modelování široké třídy problémů, jehož vlastnosti již nyní testujeme.

V budoucnu plánujeme další rozšíření třídy dopředných strukturovaných neuronových sítí a to zejména o nové typy neuronů, které by umožnili robustnější klastrování dat a nelineární optimalizaci parametrů jednotlivých neuronů. Paralelně s tímto zobecněním podrobíme dalším experimentům genetickou optimalizaci architektury neuronových sítí. V neposlední řadě pak plánujeme vytvořit analýzu teoretických vlastností těchto neuronových sítí a použitého procesu optimalizace.

References

[Gruau94] Gruau F., "Neural Network Synthesis using Cellular Encoding and The Genetic Algorithm", *Doctor Thesis, Lyon*, 1994.

[Read72] Read Ronald C., "The coding of various kinds of unlabeled trees", "Graph theory and computing", pp. 153 - 182, 1972.

[BSK95] Bitzan P., Šmejkalová J., Kučera M., "Neural networks with switching units", "Neural Network World", vol. 4, pp 515-526, 1995.

Model of Flow and Solute Transport in Dual-porosity Media

doktorand:
MILAN HOKR
Dpt. of Modelling of Processes, Faculty of Mechatronics, Technical University of Liberec, Hálkova 6, 461 17 Liberec
Dpt. of Computational Methods, Institute of Computer Science, Academy of Sciences of the Czech Republic, Pod Vodárenskou věží 2, 182 07 Prague 8
milan.hokr@vslib.cz

školitel:
JIŘÍ MARYŠKA
Department of Modelling of Processes, Faculty of Mechatronics, Technical University of Liberec, Hálkova 6, 461 17 Liberec, Czech Republic
jiri.maryska@vslib.cz

obor studia:
Přírodovědné inženýrství

Abstract

The paper deals with modelling of groundwater solute transport in dual-porosity environment, i.e. porous media with blind pores containing immobile water. We present a numerical model, composed of several parts: the fluid flow is solved by the mixed-hybrid finite elements method and the transport problem is solved by the operator splitting method, where the split problems are advection (solved by explicit upwind finite volume scheme) and the mobile–immobile exchange (expressed by analytical solution) – this combination of methods is author's result in the thesis. The model results are successfully compared with analytical solutions on simple 1D problem. The model was also applied on the real-world problem of groundwater remediation in Stráž pod Ralskem region and the material parameters in the model are calibrated by comparisons with field measurements.

1. Introduction

Numerical calculations play important role in the groundwater management for forecasting of results of artificial operation (pumping or injecting water) and for protection of drinking water sources against contaminants. Many numerical methods were developed in the last 30 years for basic problems of fluid flow and solute transport and various generalisations. The more powerful computer technology allows to model larger scale of problems and to use appropriately sophisticated methods.

The effect of dual porosity (blind pore zone with immobile water) appears in many porous media and belongs to intensively studied topics of interaction between flowing chemical substances and the solid matrix of porous media. The influence of the blind (immobile) pores in the material evinces e.g. during extraction of contaminated water – the gradual diffusive motion of solutes from the immobile to the mobile pores

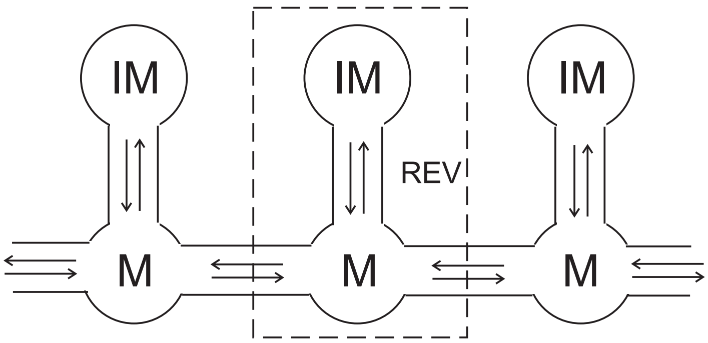


Figure 1: Structure of the two-region model. Transport connection between mobile and immobile pores at the level of REV (representative elementary volume [3]). The scheme corresponds to the communication structure in the numerical model.

(with moving water) [1] causes the remediation to take longer time, the concentration in the drawn solution decreases slower.

The topic of the thesis (construction and analysis of the model) was motivated by remediation of contaminated underground water in Stráž pod Ralskem region in the north of the Czech Republic [2], where the model is currently being used. In spite of that, the most of the presented work is general and can be applied for other porous media problems with the considered properties.

2. Problem description

The problem of transport in dual-porosity media (i.e. media with interacting mobile and immobile pores) is commonly represented by the two-region model: The concentration of solute is represented by two functions c_m and c_i, denoting the concentrations in mobile and immobile pores respectively [1]. The water flow in the mobile pores is governed by the Darcy's law. We regard such a structured porous media as two overlaying continua, interacting to each other, see Fig. 1.

The solute transport in porous media [3] is governed by the advection-dispersion equation. The mobile–immobile exchange is introduced as a source term in the equation, with the rate proportional to the concentration difference by a coefficient α. The overall process is governed by the system of equations

$$\frac{\partial c_m}{\partial t} + \nabla \cdot (c_m\, \boldsymbol{v}) - \nabla \cdot (\mathbb{D}\,\nabla c_m) = \frac{1}{n_m}\alpha(c_i - c_m) + c^* q_s^+ + c_m q_s^- \,, \tag{1}$$

$$\frac{\partial c_i}{\partial t} = -\frac{1}{n_i}\alpha(c_i - c_m)\,, \tag{2}$$

where c_m, c_i are the unknown concentrations mentioned above, $\mathbb{D}$ is the matrix of dispersion coefficients (functions of velocity, see e.g.[3]), $q_s^+ > 0$ is the fluid source intensity, $q_s^- < 0$ is the fluid sink intensity (volume per unit volume of mobile fluid and unit time), c^* is the injected solute concentration given and $\boldsymbol{v}$ is the seepage velocity.

The velocity field $\boldsymbol{v}(\boldsymbol{x})$ must be determined as a solution of the fluid flow problem [3], defined by the Darcy's Law

$$\boldsymbol{v} = -\frac{1}{n_m}\mathbb{K}\nabla\phi \tag{3}$$

and the mass balance (continuity) equation

$$\nabla \cdot \boldsymbol{v} = q_s^+ + q_s^- \tag{4}$$

where $\mathbb{K}$ is the tensor of hydraulic conductivity and ϕ is the piezometric head. We assume the flow in saturated porous media (no free water surface).

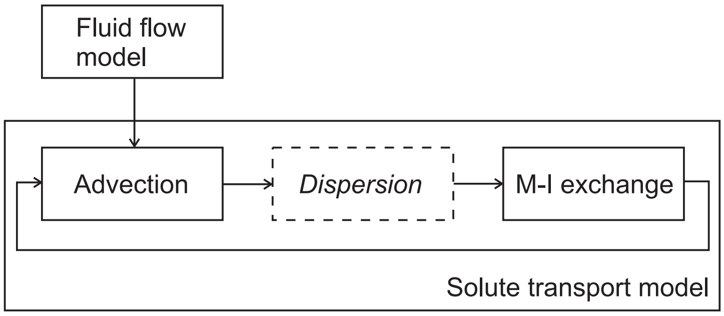


Figure 2: The scheme of the overall model structure: communication of the flow and transport model and operator splitting in the transport model.

Both the problems of flow and transport are solved in the domain $\Omega \subset \mathbb{R}^3$. Mixed Dirichlet and Neumann boundary conditions are prescribed, but the division of the boundary is different for the respective problems.

3. Numerical method

The structure of the model demonstrated in Fig. 2. We constructed the numerical method in such a way that we could conveniently separate the processes, each solved by simple and efficient method. The methods are required to be compatible with each other (the mesh and the discrete values of unknowns) and to be applicable for 3D problems with complex geometry. We also preferred to keep the structure of former models of flow and transport, taking advantage of its interface for realistic problems (unstructured mesh geometry, general choice of coefficients and boundary conditions, etc.).

We used a mixed-hybrid finite element (MH-FEM) approximation of the flow problem [4]. The discretisation of the transport by the finite volume method (FVM) benefits from a convenient coupling with the MH-FEM fluid flow, whose resulting values are fluxes through element edges, which are the input values of the transport model. The flux results of MH-FEM approximation have the important property of fluid mass balance ($U_{kj} = -U_{jk}$ below) with a consequence in the solute mass balance in the FVM transport model.

The FVM transport model is based on cell-centred representation [5] and a scheme explicit in time. The operator splitting for the non-equilibrium transport problem (1)-(2) was derived on the base of standard splitting of the advection-dispersion problem [6]. This approach was also inspired by other splitting methods for non-equilibrium problems [7, 8]. The advantage of the presented approach is the possibility to use the analytical solution of the split problem of mobile–immobile exchange (besides the other two – advection and dispersion). Other methods of solution of the solute transport with non-equilibrium mobile–immobile exchange, mostly more complicated, are discussed in [9].

We describe the operator splitting for the general case, but for the practical realisation of the model, we consider simplified problem without the dispersion term, which is often appropriate approximation for the real-world problems (advection-dominated transport and difficult identification of the dispersion parameters). Particular methods are described only for the advection and the mobile–immobile exchange.

3.1. Mixed-hybrid FEM for the fluid flow

The mixed-hybrid FEM is based on weak formulation of the equations (3)-(4) on a system $\mathcal{E}_h$ of subdomains of Ω with additional constraint on the interfaces – see [4] and references therein for details. The unknown functions ϕ, $\boldsymbol{u}$, and λ with the lowest regularity L_2, H^1 and L_2 respectively, represent the pressure, velocity and traces of pressures at element sides.

In the discrete (approximating) form, the unknowns are approximated by element-wise constant, element-wise linear and side-wise constant base functions (in the order above). The solved variables relating to velocities have a meaning of fluxes through the element sides (U_{jk} in the transport model).

The problem is given by the following system of equations for the triplet of unknowns

$$(\boldsymbol{u}_h, \phi_h, \lambda_h) \in \boldsymbol{RT}^0_{-1}(\mathcal{E}_h) \times M^0_{-1}(\mathcal{E}_h) \times M^0_{-1}(\Gamma_h) \ : \tag{5}$$

$$\sum_{e\in\mathcal{E}_h} \{(\boldsymbol{A}\boldsymbol{u}_h, \boldsymbol{w}_h)_{0,e} - (\phi_h, \nabla\cdot\boldsymbol{w}_h)_{0,e} + \langle\lambda_h, \boldsymbol{\nu}^e\cdot\boldsymbol{w}_h\rangle_{\partial e\cap\Gamma_h}\}$$
$$= \sum_{e\in\mathcal{E}_h} \langle\phi_{D,h}, \boldsymbol{\nu}^e\cdot\boldsymbol{w}_h\rangle_{\partial e\cup\Gamma_D} \quad \forall\boldsymbol{w}_h \in \vec{RT}^0_{-1}(\mathcal{E}_h)\,, \tag{6}$$

$$-\sum_{e\in\mathcal{E}_h} (\nabla\cdot\boldsymbol{u}_h, \psi_h)_{0,e} = -(q_h, \psi_h)_{0,\Omega} \quad \forall\psi_h \in M^0_{-1}(\mathcal{E}_h)\,, \tag{7}$$

$$\sum_{e\in\mathcal{E}_h} \langle\boldsymbol{\nu}^e\cdot\boldsymbol{u}_h, \mu_h\rangle_{\partial e} = \langle u_{N,h}, \mu_h\rangle_{\Gamma_N} \quad \forall\mu_h \in M^0_{-1}(\Gamma_h)\,, \tag{8}$$

where the boundary conditions are incorporated by functions $\phi_{D,h}$ and $u_{N,h}$ and by the choice of Γ_h (union of element faces except those with prescribed Dirichlet boundary condition). The notations $\boldsymbol{RT}^0_{-1}$, M^0_{-1}, M^0_{-1} denote the approximation spaces, $\boldsymbol{\nu}$ is outword normal to element boundary; see [4] for further details.

Expressing the integrals (scalar products) in the system (6)–(8) for the base functions, we obtain an equivalent system of linear algebraic equation with a specific structure.

$$\begin{pmatrix} \mathbb{A} & \mathbb{B} & \mathbb{C} \\ \mathbb{B}^T & & \\ \mathbb{C}^T & & \end{pmatrix} \begin{pmatrix} U \\ \Phi \\ \Lambda \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \end{pmatrix}. \tag{9}$$

Overall, the matrix in (9) is symmetric and indefinite (and of course sparse, as typical for the discretisations of PDEs). Methods for solving this type of system are discussed e.g. in [10]. The solver used in our computations is based on Schur-complement reduction and conjugate gradient method.

3.2. Operator splitting for the solute transport

To define the operator splitting [6], we rewrite the system of equations (1)–(2) in the operator form

$$\frac{\partial \boldsymbol{c}}{\partial t} = \mathcal{A}(\boldsymbol{c}) + \mathcal{D}(\boldsymbol{c}) + \mathcal{X}(\boldsymbol{c})\,, \tag{10}$$

where $\boldsymbol{c} = \begin{pmatrix} c_m \\ c_i \end{pmatrix}$ is the couple of unknown concentrations, the operators of advection $\mathcal{A}$, dispersion $\mathcal{D}$ and exchange $\mathcal{X}$ are defined as follows

$$\mathcal{A}(\boldsymbol{c}) = \begin{pmatrix} -\nabla\cdot(c_m\boldsymbol{v}) + c^*q^+ + c_m q^- \\ 0 \end{pmatrix}, \tag{11}$$

$$\mathcal{D}(\boldsymbol{c}) = \begin{pmatrix} -\nabla\cdot(\mathbb{D}\nabla c_m) \\ 0 \end{pmatrix}, \tag{12}$$

$$\mathcal{X}(\boldsymbol{c}) = \begin{pmatrix} \dfrac{1}{n_m}\alpha\,(c_i - c_m) \\ -\dfrac{1}{n_i}\alpha\,(c_i - c_m) \end{pmatrix}, \tag{13}$$

and the initial and boundary conditions stay in the unchanged form. Choosing a time discretisation, the processes are separated by successive solving of equations with single operators $\mathcal{A}$, $\mathcal{D}$, $\mathcal{X}$ on the right-hand side instead of (10), in each time step.

The simplified calculation with advection and mobile–immobile exchange with cell-centred values of concentration can be expressed by the values

$$\boxed{C_m^{k,n}\ ,\ C_i^{k,n}} \overset{\text{advection}}{\rightarrow} \boxed{C_m^{k,\,n+\frac{1}{2}}\ ,\ C_i^{k,\,n+\frac{1}{2}}} \overset{\text{exchange}}{\rightarrow} \boxed{C_i^{k,\,n+1}\ ,\ C_i^{k,\,n+1}}\,, \tag{14}$$

where k is cell index, m and i denote the mobile and immobile zone. We remark that $C_i^{n+\frac{1}{2}} = C_i^n$, due to no advection in the immobile pores.

3.3. Scheme for advection

We use the upwind scheme with cell-centred concentrations in the mobile zone, redenoting $C_k^n \equiv C_m^{k,n}$ for readability,

$$C_k^{n+\frac{1}{2}} = C_k^n + \frac{\Delta t}{V_k} \cdot \big[-\sum_{j \in N_k} (U_{kj}^+ C_k^n + U_{kj}^- C_j^n) + C_k^n Q_k^- + \tilde{C}_k^n Q_k^+ \big] \,, \tag{15}$$

where Δt is the time step duration, Q_k is the source/sink intensity of fluid, $\tilde{C}_k^n$ is the injected concentration (given), V_k is the volume of the cell, N_k is the index set of neighbour cells. Next, U_{kj} are the fluxes from k-th cell to j-th cell, and the superscripts $+$ and $-$ behave as a "switch" between a positive or negative number ($a^+ = a$ for $a \geq 0$ and $a^+ = 0$ for $a < 0$ etc.). Thanks to the mass balance in the MH-FEM flow model, it holds $U_{kj} = -U_{jk}$.

The stability condition for the scheme is

$$\Delta t \sum_{j \in N_k} U_{kj}^+ \leq V_k \qquad \text{and} \quad \Delta t \sum_{j \in N_k} (-U_{kj}^-) \leq V_k \qquad \forall k \,, \tag{16}$$

which is a form of the well-known CFL condition in 1D, $C_r = \frac{v \Delta t}{\Delta x} \leq 1$, where C_r is the Courant number. This is a restrictive condition on the time step, which must not exceed certain value, depending on the mesh and the velocity field. The practical compliance of the condition is performed by successive halving of user-given time step, until the condition (16) in all the discretisation volumes is fulfilled.

3.4. Scheme for non-equilibrium exchange

The operator splitting in the system (1)-(2) leads to a solution of two coupled ODE in a given space point (or discretisation volume) in each time step

$$\frac{\mathrm{d}c_m}{\mathrm{d}t} = \frac{1}{n_m} \alpha (c_i - c_m) \,, \tag{17}$$

$$\frac{\mathrm{d}c_i}{\mathrm{d}t} = -\frac{1}{n_i} \alpha (c_i - c_m) \,, \tag{18}$$

which honour the mass balance $n_m c_m + n_i c_i = \mathrm{const}$. The exact solution of (17)-(18) for arbitrary time $t \geq 0$ is

$$c_m(t) = \big(c_m^{(0)} - \bar{c}^{(0)}\big) \mathrm{e}^{-\alpha \frac{n_m + n_i}{n_m n_i} t} + \bar{c}^{(0)} \tag{19}$$

and by analogy for $c_i(t)$. We denoted $c_m^{(0)}$ and $\bar{c}^{(0)} = \frac{n_m c_m + n_i c_i}{n_m + n_i}$ the initial values for $t = 0$. Next, we denote $\tilde{\alpha} = \alpha \frac{n_m + n_i}{n_m n_i}$ the modified exchange coefficient and the characteristical time of exchange $T_{1/2} = \frac{\ln 2}{\tilde{\alpha}}$, which is the value used in practice as input value of the model. The discretisation scheme for both mobile (m) and immobile (i) concentrations is

$$C_{i,m}^{n+1} = (C_{i,m}^{n+\frac{1}{2}} - \overline{C}^{n+\frac{1}{2}}) \mathrm{e}^{-\tilde{\alpha} \Delta t} + \overline{C}^{n+\frac{1}{2}} \,, \quad \overline{C}^{n+\frac{1}{2}} = \frac{n_m C_m^{n+\frac{1}{2}} + n_i C_i^{n+\frac{1}{2}}}{n_m + n_i} \,, \tag{20}$$

where we omitted a symbol for mesh cell k for readability.

The expression (20) can be used for any time step, thus, comparing with the advection term alone, there is no more requirement on the time step of the model.

4. Solution of example problems

The example problems were solved to verify a correctness of the used numerical scheme and to discuss the significance of the numerical error in the context of solving hydrogeological problems.

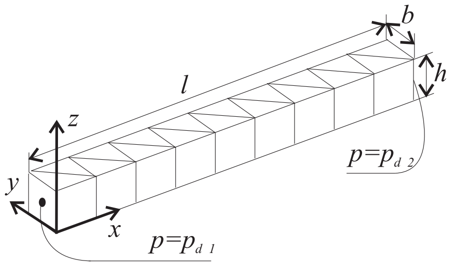


Figure 3: The mesh for example calculation of 1D transport with 3D model

The problems below are "1D" and "2D" in the sense of problem symmetry; with the 3D model, they are solved in a 3D domain, with the rest dimensions given by a discretisation step, see e.g. Fig. 3.

4.1. Handling the numerical dispersion in comparisons

The upwind scheme for advection brings the numerical dispersion, which significantly changes the results compared with the exact solution of the advection. For our comparisons, we included the numerical dispersion into a considered analytical solution to separate that numerical error (expected) from the possible error of the discretisation of the mobile–immobile exchange and the operator splitting.

The numerical dispersion in 1D can be expressed [11] by a dispersion coefficient

$$D_{num} = \frac{1}{2} v \Delta x (1 - C_r)\,, \tag{21}$$

where Δx is the space discretisation step.

In practice, the above formula can be used for a-posteriori estimating of the numerical dispersion. If we find that the value is too high in comparison with the real physical dispersion (we consider an advection-dominated problem), it is necessary to construct a finer mesh for such a problem.

4.2. Transport in uniform 1D flow

We solve a problem of transport in uniform flow with zero initial condition (both concetration c_m and c_i) and constant value of concentration at the inflow point. The parameters are the following:

length of the domain	$L = 1000\mathrm{m}$
velocity	$v = 1\mathrm{m/day}$
porosities	$n_m = 0.1\; n_i = 0.2$ or $n_m = 0.2\; n_i = 0.1$
input concentration	$c_0 = 100\mathrm{kg/m^3}$
observed time interval	$t_{\max} = 500\mathrm{days}$

where we consider two choices of porosities with opposite weighting.

The calulations were performed for the rates of the mobile–immobile exchange covering the full possible scale, from 0 to ∞. The values are in Tab.1 in terms of the characteristic time of exchange and the dimensionless form of the coefficient.

4.2.1 Numerical solution: A natural discretisation of the 1D canal problem using the trilateral prisms is by pairs in blocks, Fig. 3. The 1000m domain is discretised into 20 blocks, i.e. 40 trilateral volumes, with their centers uniformly spaced: $\Delta x = 25\mathrm{m}$. The time step is chosen $\Delta t = 12.5\mathrm{days}$, thus the Courant number is $C_r = \frac{1}{2}$.

Using (21), we express the numerical dispersion in this problem. The equivalent dispersion coefficient in the advection-dispersion equation is $D = 6.25\mathrm{m}^2$/day.

Table 1: Values of $T_{1/2}$ chosen for the example calculations and comparison of numerical and analytical solutions. The values of ω are appropriately derived (23) for the use in the analytical solution.

	$n_m = 0.1, n_i = 0.2$	$n_m = 0.2, n_i = 0.1$
$T_{1/2}$ [days]	ω (dimensionless)	
∞ (turned-off exchange)	0	0
1000	0.462	0.231
100	4.62	2.31
10	46.2	23.1
0 (equilibrium)	∞ (not calculated)	∞

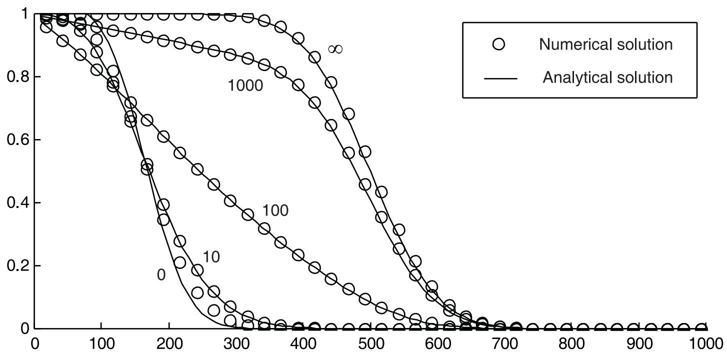


Figure 4: Comparison of numerical and analytical solution for the 1D non-equilibrium transport problem, for the porosities $n_m = 0.1$, $n_i = 0.2$. The labels denote the values of exchange time $T_{1/2}$.

4.2.2 Analytical solution: We consider the analytical solution presented in [12], which is expressed by a formula of several lines. For reference, we use the same notation of their dimensionless variables

$$X = \frac{x}{L}, \quad T = \frac{vt}{L}, \quad \beta = \frac{n_m}{n_m + n_i}, \tag{22}$$

$$P = \frac{vL}{D} \text{ (Peclet number)} \quad \text{and } \omega = \frac{\alpha L}{n_m v}\,. \tag{23}$$

In our problem, the coefficients are $\beta = \frac{1}{2}$, $P = 160$ and ω in Tab.1.

4.2.3 Discussion: Both the numerical and the analytical solution are displayed in the Fig. 4 for the first choice of the porosities. Results for the second variant are similar, but the “retardation” in the equilibrium case ($T_{1/2} = 0$) is smaller: one third instead of two thirds as given by the ratio of porosities.

The results fit well. We can observe that the approximation of the mobile–immobile exchange does not bring other significant numerical error besides the numerical dispersion of the scheme for the advection.

4.3. Transport in 2D radial flow

This model problem was chosen to represent a typical situation in the hydrogeological actions – drawing or injecting well. We consider a circle domain of radius R with the well in the center. For correct dimensions and for the use of the 3D model, we consider thickness h.

To avoid problems with singular point in the circle center, we represent the well as a small circle with radius r_w, applying the side flux boundary condition instead of a sink. We used the values

$$R = 100\text{m},\ r_w = 5\text{m}, h = 10\text{m},\ Q = 48\text{m}^3/\text{d}\,,$$

where Q is the injected/drawn rate.

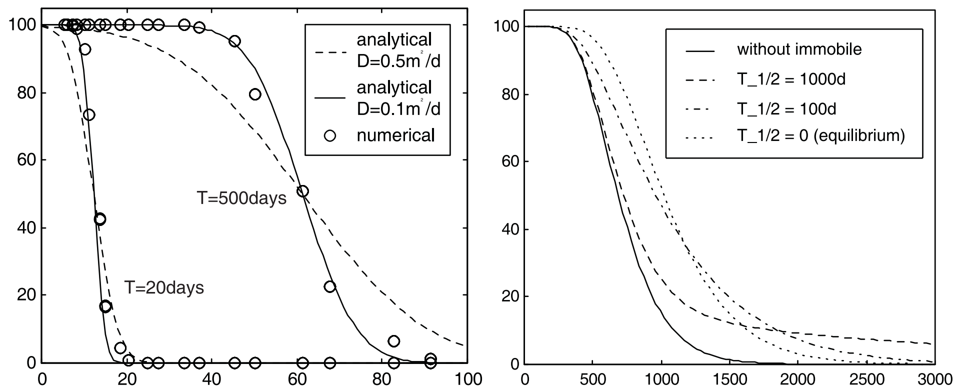


Figure 5: Solution of transport problems in 2D radial flow. In the left: Concentration vs. position in the radial direction (meters) for the problem of injecting. In the right: Concentration vs. time (days) for the problem of drawing.

Two problems were solved: First, the injection of constant concentration to the domain with zero initial state, observing only the behavior of the advection model. Second, the extraction from the domain with non-zero constant concentration at the beginning and zero-concentration in the inflow from the outside.

4.3.1 Numerical solution: The mesh was constructed as non-uniform, finer close to the well and coarser at the outside boundary. The dimensions in the radial direction are approximately proportional to the position, $\Delta x \sim r$, from 1.75m to 25m. Thanks to the radial symmetry, we solved the problem in a $\frac{\pi}{4}$-sector, reducing the total number of cells to 140.

The numerical dispersion (in radial, i.e. longitudinal direction) estimated by (21) is expressed by almost constant dispersion coefficient $D = 0.5\mathrm{m}^3$/day ($C_r \approx 0$ in most of the cells, except close to the well and v is inverse proportional to r).

4.3.2 Analytical solution of advection and dispersion: We did not find an analytical solution of the general non-equilibrium transport problem. For a basic check of the advection and the numerical dispersion, we use the classical solution of 1D advection-dispersion problem, expressed by $\mathrm{erfc}\left(\frac{x-vt}{\sqrt{4Dt}}\right)$, with appropriately placing the "wave front" according to the non-uniform velocity.

In comparison for a given time (Fig. 5 in the left) we can observe that the numerical result fit to the analytical with the dispersion coefficient $D = 0.1\mathrm{m}^2$/d instead of the estimated numerical dispersion $D = 0.5\mathrm{m}^2$/d. It can be caused by our many simplification in calculation of both the results.

4.3.3 Influence of mobile–immobile exchange: We compared (Fig. 5 in the right) the time-dependences of the drawn concentration for various values of mobile–immobile exchange coefficient $T_{1/2}$. The quicker mobile–immobile exchange takes effect in the earlier part of the time interval, while the slower exchange takes effect in the later time, as expected.

5. Application of the model

5.1. Problem description

The problem we present here is a subset of the global model of remediation control. There are many of the local fields like the one presented and the remediation process is handled by weighted drawing from all these fields.

We consider a single leeching field with the horizontal dimesions about 1200×500m and $60 - 90$m thickness, covered by the mesh of 12000 cells. The time interval is 18 months and it begins at the beginning of remediation drawing. There are 20 drawing wells in the area. The period solved is splitted into one-month

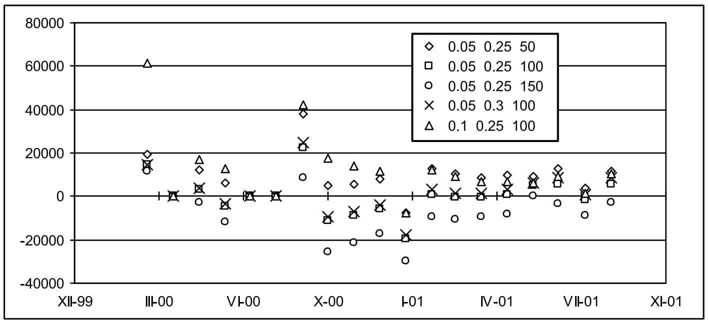


Figure 6: Deviations of the total drawn mass (kg) in single months, comparing the numerical results and field measurements. Results for various sets of material parameters (in the order: mobile porosity n_m, total porosity $n_{\mathrm{tot}} = n_m + n_i$, and characteristic time of mobile–immobile exchange $T_{1/2}$).

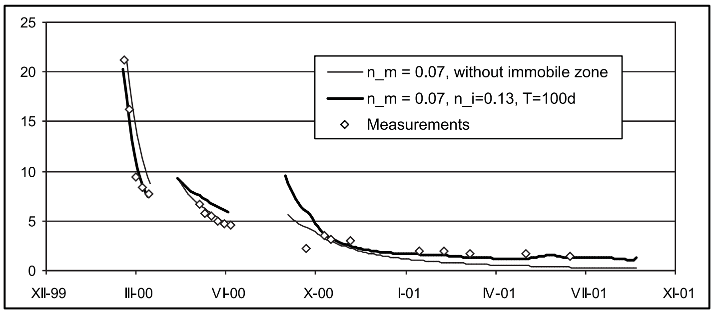


Figure 7: Drawn concentration (m^3/kg) versus time (dates) for a selected well. Comparison of the model without immobile pores, model with calibrated parameters, and measurements.

intervals: there are given constant flow rates in the wells in each month and measured the total mass of drawn solutes.

5.2. Discussion of the results

The model with mobile–immobile exchange showed expected hehavior: the concentration in the drawn solution decreases slower, mainly in the later period, when the model without the immobile pores forecasts almost zero concentration.

The calculations were performed for a set of values n_m, n_i and $T_{1/2}$, which are not sufficiently determined from laboratory soil analyses. Comparing the deviations of the numerical result and measurements helped to specify the values more precisely, but on the other hand, the deviations change with time (Fig. 6), which makes the calibration difficult and non-unique. The comparions were done on global measure: the sum of solute mass from all the wells.

Comparisons of concentrations in the single wells were used for secondary validation, because there were often deviations of both signs among all the wells and calibrating one well correctly caused the rest wells to have worse fitting results. An example of the results is in Fig. 7, where we can observe good agreement in the final period, but worse agreement in the middle period.

References

[1] K. H. Coats and B. D. Smith, "Dead-end pore volume and dispersion in porous media," *Soc. Pet. Eng. J.*, vol. 4, pp. 73–84, 1964.

[2] J. Novák, "Groundwater remediation in the Stráž leaching operation," *Mine Water and the Environment*, vol. 20, pp. 158–169, 2001.

[3] J. Bear and V. Verruijt, *Modeling groundwater flow and pollution*. Dordrecht, Holland: D. Reidel, 1990.

[4] J. Maryška, M. Rozložník, and M. Tůma, "Mixed-hybrid finite-element approximation of the potential fluid-flow problem," *J. Comput. Appl. Math.*, vol. 63, pp. 383–392, 1995.

[5] C. Hirsch, *Numerical Computation of Internal and External Flows, Volume 1 – Fundamentals of Numerical Discretization*. John Wiley & Sons Ltd., 1991.

[6] M. Crandall and A. Majda, "The method of fractional steps for conservation-laws," *Numerische Mathematik*, vol. 34, no. 3, pp. 285–314, 1980.

[7] J. van Kooten, "A method to solve the advection-dispersion equation with a kinetic adsorption isotherm," *Adv. Water Resour.*, vol. 19, no. 4, pp. 193–206, 1996.

[8] J. Kačur and P. Frolkovič, "Semi-analytical solutions for contaminant transport with nonlinear sorption in 1D." IWR/SFB preprint, University of Heidelberg, 2002.

[9] C. Gallo, C. Paniconi, and G. Gambolati, "Comparison of solution approaches for the two-domain model of nonequilibrium transport in porous media," *Adv. Water Resour.*, vol. 19, pp. 241–253, 1996.

[10] J. Maryška, M. Rozložník, and M. Tůma, "Schur complement systems in the mixed-hybrid finite element approximation of the potential fluid flow problem," *SIAM J. Sci. Comput.*, vol. 22, pp. 704–723, 2000.

[11] C. Zheng and G. D. Bennett, *Applied contaminant transport modeling*. New York: Van Nostrand Reinhold, 1995.

[12] N. Toride, F. Leij, and M. van Genuchten, "A comprehensive set of analytical solutions for nonequilibrium solute transport with first-order decay and zero-order production," *Water Resour. Res.*, vol. 29, pp. 2167–2182, 1993.

Analýza rozhodovacích lesů

doktorand:
EMIL KOTRČ
Ústav informatiky AV ČR, Pod Vodárenskou věží 2, Praha 8
kotrc@cs.cas.cz

školitel:
RNDR. PETR SAVICKÝ, CSC.
Ústav informatiky AV ČR, Pod Vodárenskou věží 2, Praha 8
savicky@cs.cas.cz

obor studia:
Matematické inženýrství

Abstrakt

Rozhodovací stromy a lesy jsou jedněmi ze základních metod strojového učení a ve srovnání s ostatními dosahují velmi dobrých výsledků. Zároveň jsou tyto metody poměrně jednoduché a velice rychlé, jak při konstrukci, tak při vyhodnocování. Tento článek je věnován jedné poměrně nové a zajímavé metodě zvané *Random Forest*. Na konkrétních datech bude ukázáno, jak lze lesy zkonstruované pomocí *Random Forest* analyzovat a odvozovat jejich chování pro různá pravidla kombinování jednotlivých stromů v lese zvolené velikosti; to vše pouze ze znalosti chování jednoho konkrétního velkého lesa. Hlavním cílem tohoto příspěvku je popsat nástroj, který umožní hledání optimálních vah stromů v lese, jejichž použití by vedlo ke zlepšení predikce.

1. Úvod

V tomto článku se budeme zabývat analýzou rozhodovacích lesů, které vytváří metoda *Random Forest* [4]. Rozhodovací lesy jsou společně s rozhodovacími stromy jednou z metod strojového učení. V příspěvku se budeme těmito metodami zabývat z pohledu klasifikace, tedy budeme se zabývat analýzou klasifikátorů ve formě rozhodovacích lesů. Klasifikátor je obecně nějaké zobrazení $h : \mathbf{X} \to C$, kde $\mathbf{X}$ je nějaký prostor vektorů a C je množina možných tříd, do kterých můžeme prvky z $\mathbf{X}$ zařadit. Pro konstrukci klasifikátoru se ve většině případech využívá znalosti takzvané učící množiny, což je nezávislý výběr případů se známou klasifikací z prostoru $\mathbf{X}$.
Rozhodovací strom je tedy nějaký klasifikátor h, který má tvar stromu se dvěma typy uzlů – rozhodovací uzly, obsahující nějaký test, a hrany do dalších uzlů; a listy, které nějakým způsobem indikují třídu. Neznámý případ, který chceme oklasifikovat, tak prochází jednotlivými testy v rozhodovacích uzlech až dosáhne nějakého listu, jenžto určí jeho třídu.
Rozhodovací les potom není nic jiného než množina rozhodovacích stromů a pravidlo pro kombinaci predikcí jednotlivých stromů. V současné době existuje několik metod a implementací na konstrukci rozhodovacích stromů a lesů – *C4.5* a *C5.0* [1, 3], *CART* [2], *CRUISE, QUEST* [7, 8] a *Random Forest* [4].
Zde se budeme zabývat metodou *Random Forest* a zejména vhodnou volbou vážení jednotlivých stromů v lese. Ukážeme jak lze snadno lesy z *Random Forest* analyzovat a odvozovat jejich chování ze znalosti

chování velkého lesa, což bude mít význam pro hledání optimálních vah.
Naším výsledkem je tedy nástroj respektive postup, který nám umožní pouze ze znalosti predikcí a vlastností stromů v jednom konkrétním velkém lese (zvolili jsme les velikosti 500 stromů) odvodit *průměrné* chování lesa zvolené velikosti při použití různých pravidel pro kombinaci stromů. Důležité je, že odvozujeme průměrné chování lesa. Místo konkrétního lesa, který by mohl být zatížen chybou, totiž používáme statistický model, který je určen parametry odhadnutými z již zmiňovaného lesa o 500 stromech. V článku zároveň ukážeme i jeden konkrétní případ vah (pravidlo pro kombinování stromů), který při daném počtu stromů v lese klasifikaci zlepší.

1.1. Data

Většinu experimentů provádíme na datech z projektu *MAGIC-telescope*[1]. Jedná se o data s 13 numerickými prediktory a se dvěma třídami $\mathbf{-1}, \mathbf{+1}$ (šum a signál), přičemž na učení se používá prvních 10 atributů $(x_1, \ldots, x_{10})$. Označme si $\mathbf{X}$ množinu všech možných měření a $C = \{-1, +1\}$ množinu možných klasifikací. Pro naučení klasifikátoru budeme potřebovat učící množinu, nechť $l = \{\mathbf{x}_1, \ldots, \mathbf{x}_S\}$, kde $\mathbf{x}_i \in \mathbf{X}$, $\mathbf{x}_i = (x_1, \ldots, x_{10}), i \in \{1, \ldots S\}$ je pro nás množina vektorů se známou klasifikací. Tyto známé klasifikace můžeme vyjádřit pomocí čísel $y_i \in C; i \in \{1, \ldots, S\}$; tak, že pokud je například vektor $\mathbf{x}_i$ prvkem třídy $\mathbf{-1}$, pak $y_i = -1$. Učící množina pak pro nás bude množina dvojic $\mathbf{z}_i = (\mathbf{x}_i, y_i)$; $L = \{\mathbf{z}_1, \ldots, \mathbf{z}_S\}$. Na základě této množiny pak můžeme pomocí *Random Forest* vygenerovat klasifikátor ve formě rozhodovacího lesa.
Podobným způsobem jako učící množinu můžeme definovat testovací množinu $K = \{\mathbf{z}_1, \ldots, \mathbf{z}_M\}, \mathbf{z}_i = (\mathbf{x}_i, y_i)$. S využitím této množiny pak budeme porovnávat a provádět analýzu vzniklých lesů. Označme si ještě navíc M_+ počet případů z K, které patří do třídy $\mathbf{+1}$ a podobně M_- jako počet případů z $\mathbf{-1}$.
Více o datech a dřívějších experimentech, které jsme prováděli s využitím *C5.0* a *CART* lze najít v [10] a [9].

2. Random Forest

Random Forest je poměrně nová a zajímavá metoda, jejímž autorem je L.Breiman [4]. Narozdíl od metod *C4.5*, *C5.0* a *CART*, které vznikly jako metody na konstrukci jednotlivých stromů, metoda *Random Forest* je od počátku určena na generování velkých lesů. Algoritmus této metody zde nebudeme popisovat, podrobnosti lze nalézt ve článku [4]. Nicméně si zde uvedeme alespoň základní myšlenku. Při konstrukci jednotlivých stromů v lese, využívá *Random Forest* takzvaného *baggingu bez vracení*. Ten spočívá v tom, že při každém kroku (tedy při vytváření každého stromu) se provede z trénovací množiny náhodný výběr případů (bez vracení). Následně je na základě tohoto výběru vygenerován strom (metodou rozděl a panuj) s čistými listy, který není prořezáván, narozdíl od metod *C5.0* či *CART*. Zajímavý v metodě *Random Forest* je výběr testů v rozhodovacích listech při učení. Metoda vybere k náhodných proměnných, dle kterých by se v daném listě mohl provádět test a na základě podobného kriteria jako v *C4.5* provede výběr nejlepšího testu. Zajímavé je, že i pokud se k zvolí rovno 1, výsledky nejsou špatné viz.[4] . Při klasifikaci pak *Random Forest* využívá *většinového hlasování*, kdy výsledek klasifikace se řídí většinou hlasů jednotlivých stromů; jinými slovy, všechny stromy v lese mají stejnou váhu.

3. Odvození chování lesů

V této části stručně popíšeme jak lze provádět analýzu rozhodovacích lesů (z *Random Forest*). Naše snaha byla pokusit se pouze ze znalosti velkého lesa odvodit průměrnou chybu lesů různých velikostí při užití různých kombinací stromů, což by nám pomohlo při hledání optimálních vah a umožnilo by například optimalizaci vážící funkce. Z dřívějších experimentů (viz. [9]) máme totiž zkušenosti, že vhodné vážení stromů v lese může klasifikaci zlepšit.
Celou analýzu jsme rozdělili na tři případy. V prvním případě necháme jednotlivé stromy hlasovat pouze dvouhodnotově $(+1/-1)$, tedy přesně tak jak to provádí *Random Forest*. Ve druhém případě provedeme aproximaci takového hlasování pomocí normálního rozdělení, abychom se připravili na poslední situaci,

[1] http://hegra1.mppmu.mpg.de/MAGICWeb/

kde už budou hlasy jednotlivých stromů reálná čísla, tedy nějaké obecné váhy.

3.1. Případ I, hlasování $-1/+1$

Mějme nyní testovací množinu K a množinu N rozhodovacích stromů $F = \{T_1, \ldots, T_N\}$. Pro každý případ $(\mathbf{x}_i, y_i) \in K$ označme hlas j-tého stromu jako $T_j(\mathbf{x}_i) \in C$.
Předpokládejme nyní, že pro každý případ $(\mathbf{x}_i, y_i)$ z testovací množiny K známe parametry rozdělení hlasování stromů v lese – střední hodnotu μ_i a rozptyl σ_i^2, které jsme odhadli z již zmiňovaného velkého lesa. Mezi těmito dvěma parametry platí tento základní vztah

$$\sigma_i^2 = 1 - \mu_i^2 \tag{1}$$

který lze snadno odvodit z definice rozptylu a ze skutečnosti, že hlasování nabývá pouze hodnot $-1/+1$.
Označme si nyní součet hlasů jednotlivých stromů jako

$$g(\mathbf{x}_i) = \sum_{j=1}^{N} T_j(\mathbf{x}_i) \tag{2}$$

a hlasování celého lesa je potom

$$f(\mathbf{x}_i) = I(g(\mathbf{x}_i) > t) \tag{3}$$

kde funkce I je definována takto $I(true) = +1, I(false) = -1$ a t je nějaký parametr (mez) pro přijetí, například při většinovém hlasování je $t = 0$. V obecném případě (při hlasování $-1/+1$) může být $t \in \langle -N, +N \rangle$. Ze znalosti parametrů rozdělení a s využitím vztahu (1) můžeme dále odvodit pravděpodobnost, že j-tý strom bude hlasovat, že i-tý případ patří do třídy $+\mathbf{1}$

$$P(T_j(\mathbf{x}_i) = 1) = 1/2 \cdot (\mu_i + 1) \tag{4}$$

Díky tomu, že hlasování stromů v tomto případě nabývá pouze dvou hodnot, můžeme odvodit pravděpodobnost, s jakou celý les oklasifikuje i-tý případ jako $+1$ při daném t (označme si (4) jako p_i)

$$P(f(\mathbf{x}_i) = 1) = \sum_{k=[1/2\cdot(N+t)]}^{N} \binom{N}{k} p_i^k (1-p_i)^{N-k} \tag{5}$$

kde dolní mez k vlastně odpovídá počtu stromů, které při daném t hlasují jako 1.
Vyjádření chyby a srovnání lesů různých velikostí s různými pravidly kombinování provádíme s využitím takzvaných *ROC (Receiver Operator Curve)* křivek. Tyto křivky lze definovat následujícím vztahem

$$roc = \{(r,s) \in \langle 0,1 \rangle^2 | r = P(f(\mathbf{x}) = 1 | y = -1), s = P(f(\mathbf{x}) = 1 | y = 1)\}, (\mathbf{x}, y) \in K \tag{6}$$

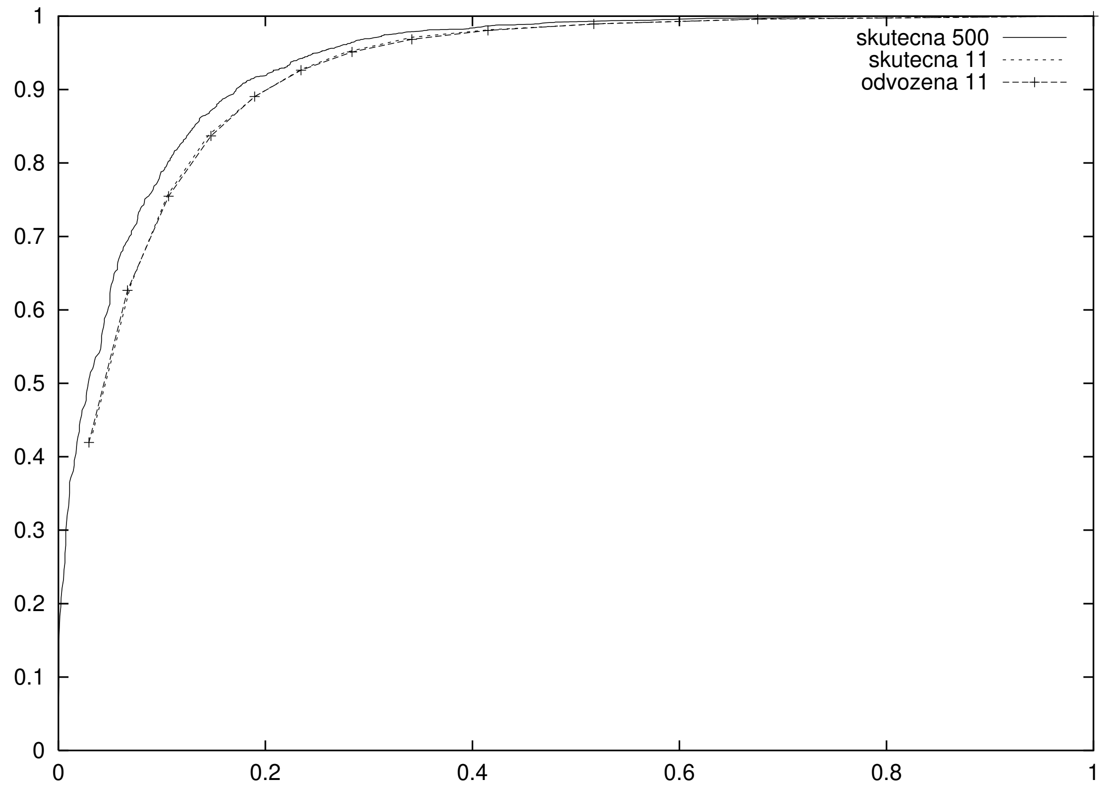
Tedy osa x vyjadřuje pravděpodobnost, že testovací případy ze třídy $-\mathbf{1}$ se přiřadí do třídy $+\mathbf{1}$, a osa y představuje pravděpodobnost s jakou jsou případy ze třídy $+\mathbf{1}$ klasifikovány správně. Ze vztahu (6) je zároveň zřejmé, že optimálním bodem je bod $(0,1)$ (všechny případy jsou správně klasifikovány).

My jsme schopni pro každý práh t s využitím (5) určit jeden bod na ROC křivce. K tomu ovšem potřebujeme vyjádřit podmíněné pravděpodobnosti $P(f(\mathbf{x}) = 1|y = c)$, $c \in \{-1, +1\}$. Ty lze s využitím (5) vyjádřit pomocí následujících vztahů

$$P(f(\mathbf{x}) = 1|y = -1) = 1/M_- \sum_{i:y_i=-1} P(f(\mathbf{x}_i) = 1) \tag{7}$$

$$P(f(\mathbf{x}) = 1|y = 1) = 1/M_+ \sum_{i:y_i=+1} P(f(\mathbf{x}_i) = 1) \tag{8}$$

Na obrázku 1 vidíme srovnání grafů (ROC křivek), které vzniknou z lesa o 500 stromech (což je vlastně náš odhad rozdělení) a z lesa o 11 stromech, a odhadem, který jsme dostali ze vzorců (5), (8) a (7). Po odvození chování lesů v nejjednodušším případě se dostaneme ke zobecnění.



Obrázek 1: Odhad chování lesa o 11 stromech

3.2. Případ II, aproximace normálním rozdělením

Předchozí případ nás poměrně podstatně omezoval tím, že umožňoval hlasování pouze dvouhodnotové. V budoucnu ale budeme potřebovat nějaké obecné váhy, tedy kdy hlasy jednotlivých stromů jsou reálná čísla. Omezení v předchozím případě je tedy dáno tím, že je tam užito binomického rozdělení a počítá se tedy pouze se dvěma možnými způsoby hlasování. Proto jsme si potřebovali připravit půdu a umožnit obecné hlasování. Tudíž jsme místo binomického rozdělení použili aproximaci normálním rozdělením. Pomocí normálního rozdělení můžeme odhadovat chování lesů při obecném hlasování; a prvním krokem je právě tato aproximace.

Předpokládejme nyní, že máme opět N stromů, jako v předchozím případě, které hlasují opět dvouhodnotově, tedy $+1/-1$; a rovněž předpokládejme, že známe parametry rozdělení μ_i a σ_i^2. Ovšem v tomto případě již obecně (počítáme s obecnými vahami) neplatí vztah (1), tudíž neplatí ani (4). Nicméně můžeme určit

$$P(f(\mathbf{x}_i) = 1) = P(g(\mathbf{x}_i) > t) = 1 - P(g(\mathbf{x}_i) \leq t) \tag{9}$$

Abych mohli tuto hodnotu určit, musíme nejdříve zjistit $P(g(\mathbf{x}_i) \leq t)$. Jak je výše uvedeno, budeme zde aproximovat hlasování lesa pomocí normálního rozdělení. Potřebujeme tedy znát parametry rozdělení hlasování celého lesa; ty ovšem dokážeme snadno zjistit ze znalosti parametrů rozdělení hlasování stromů a díky tomu, že hlasování stromů je nezávislé, protože stromy sami osobě jsou konstruovány nezávisle. Máme tedy N nezávislých stejně rozdělených náhodných veličin. Potom lze parametry rozdělení hlasování lesa určit ze vztahů

$$M_i = E(\sum_{j=1}^{N} T_j(\mathbf{x}_i)) = N \cdot \mu_i \tag{10}$$

$$S_i^2 = E(\sum_{j=1}^{N} T_j(\mathbf{x}_i) - E(\sum_{j=1}^{N} T_j(\mathbf{x}_i)))^2 = N \cdot \sigma_i^2 \tag{11}$$

pro každé $\forall i \in \{1, \ldots, M\}$. Máme tak parametry normálního rozdělení – střední hodnotu M_i a rozptyl S_i^2. Můžeme tudíž určit

$$P(g(\mathbf{x}_i) \leq t) = \frac{1}{S_i \cdot \sqrt{2\pi}} \int_{-\infty}^{t} e^{\frac{-(x-M_i^2)}{2 \cdot S_i^2}} dx \tag{12}$$

a díky tomu už spočteme (9). Můžeme tudíž určit pravděpodobnost, že les vrátí pro konkrétní případ třídu $+\mathbf{1}$ a lze tak snadno ze vztahu (6) vyjádřit ROC křivku klasifikace lesa.
Jak dopadne odhad ROC křivky s využitím normálního rozdělení oproti odhadu s binomickým rozdělením lze vidět v grafu 2. V tuto chvíli máme tedy připraven postup pro odhad chování lesa s využitím nějakých obecných reálných vah. V dalším kroku se podíváme jak dopadne použití konkrétního typu vah.

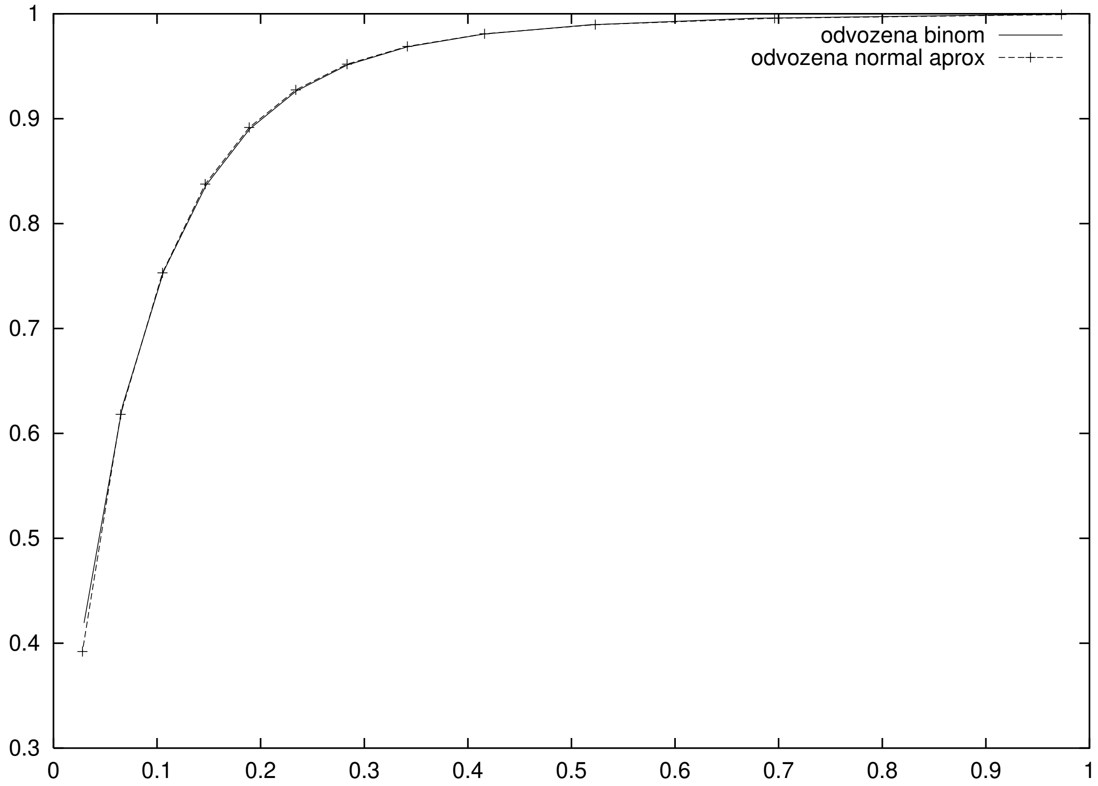
3.3. Případ III, hlasování *ks*(α), normální rozdělení

V předchozích dvou krocích jsme si vlastně pouze připravovali postup na odvození chyby při hlasování lesa s využitím obecných reálných vah. V tomto bodě se podíváme na jeden konkrétní případ vah, na kterých jsme výše uvedený postup vyzkoušeli a které jsme již dříve testovali po experimentální stránce.
Po vytvoření lesa, můžeme v každém listu každého stromu zjistit počet případů z konkrétní třídy, které se při učení do daného listu dostali. Mějme tedy list u a označme si počet případů z učící množiny z třídy $+\mathbf{1}$, které se do listu u dostali při učení jako

$$pos_u = |\{(\mathbf{x}, y) \in L | \mathbf{x} \in u \wedge y = +1\}| \tag{13}$$

Podobně můžeme pro list u definovat i neg_u – počet negativních případů (z třídy $-\mathbf{1}$). Pro každý list tak máme statistiky pos a neg. Na základě těchto dvou hodnot jsme přiřadili každému listu nějakou váhu, která závisí jednak na většině případu z nějaké třídy a rovněž na velikosti ($pos + neg$) listu. Tuto váhu listu u jsme definovali takto (ks = koeficient signifikance)

$$ks_u(\alpha) = \frac{pos_u - neg_u}{(pos_u + neg_u)^{\alpha}} \tag{14}$$



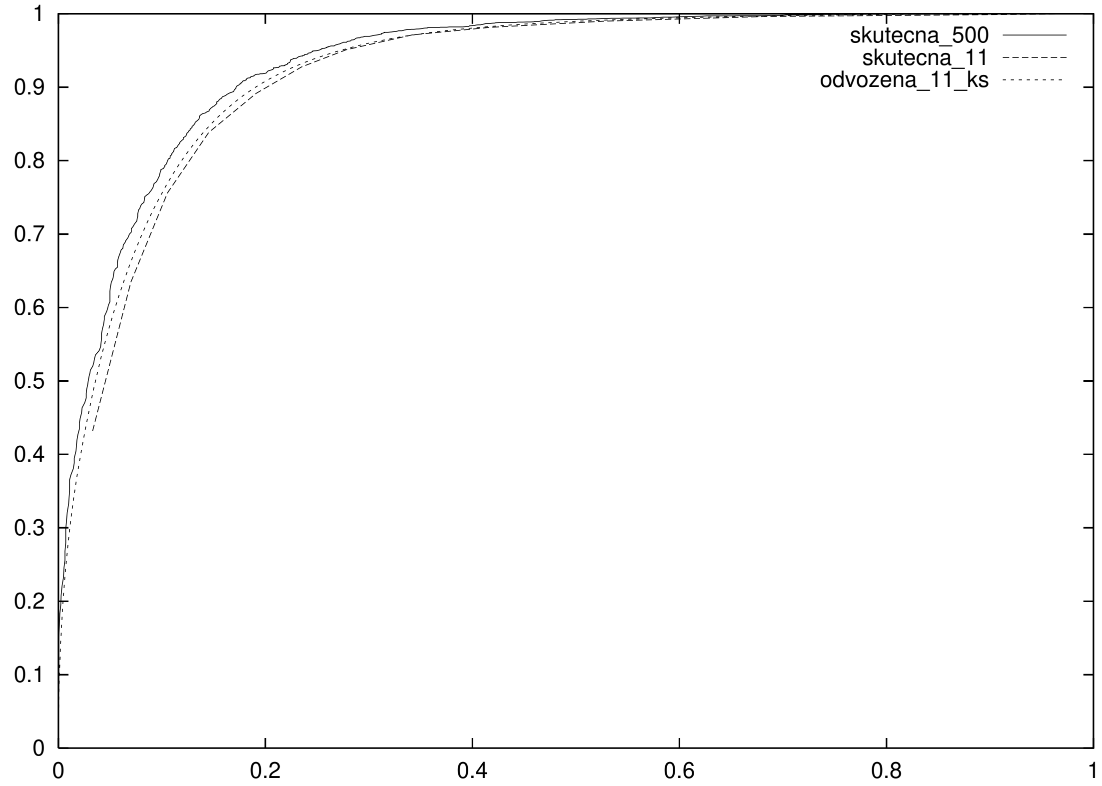
Obrázek 2: Porovnání aproximace normálním rozdělením s rozdělením binomickým

Hodnotu (14) tedy spočteme pro každý list v lese a hlasování lesa pro daný případ bude součet hodnot (14) listů, do kterých se neznámý případ "prosypal". Nyní bychom tedy rádi určili chybu při hlasování lesa s těmito vahami. Předtím než jsme měli výše popsaný postup pro odvozování chyby, museli jsme prostě skutečně vygenerovat les, na kterém jsme chybu spočítali. Pro použití našich nástrojů ovšem stačí znát parametry rozdělení μ_i a σ_i^2, tak jak jsou popsány ve druhém případě. Při prvních experimentech s vahami (14) jsme tyto parametry prostě opět odhadli z velkého lesa. To je ovšem pro budoucí práci nepoužitelné, protože bychom pro každé nové váhy museli znovu parametry odhadovat z velkého lesa. Ovšem je asi zřejmé, že i pozdější váhy se budou určovat z hodnot pos a neg, proto nám bude stačit zjistit si informace o rozdělení těchto hodnot pro každý případ na velkém lese. Toto se provede pouze jednou a bude to možno použít vícekrát, pro různé váhy. V současnosti na tomto postupu pracujeme.

Vraťme se ale zpět; předpokládejme nyní, že máme nějakým způsobem odhadnuty parametry μ_i a σ_i^2 při hlasování pomocí (14). Na základě těchto znalostí opět můžeme určit (9) a ROC křivku (6), přesně dle postupu uvedeného ve druhém případě.
Jak dopadne klasifikace s využitím hlasování (14) na 11 stromech můžeme vidět na obrázku 3; jak vidno toto hlasování klasifikaci oproti hlasování bez vah zlepší.

4. Závěr

Je zřejmé, že v tuto chvíli máme v podstatě pouze pohodlný nástroj, který nám umožní snadno analyzovat rozhodovací lesy. Jeho význam je dán tím, že při hledání a testování různých vah nemusíme generovat lesy, což bývá časově i paměťově velice náročné, a místo toho stačí v podstatě dosadit do našich vzorečků a odhadnout tak chybu lesa dané velikosti. Ukázali jsme rovněž na příkladu vah $ks(\alpha)$, že vážení může pomoci. Do budoucna se tak můžeme zabývat hledáním vhodných vah, případně hledáním bodů, které



Obrázek 3: Odhad hlasování při použití vah ks(0.7)

jsou nějakým způsobem zajímavé (například při vážení se zlepší jejich klasifikace), tedy obecně analýzou rozhodovacích lesů. V tomto článku jsme popsali nástroj, který nám toto elegantně a snadno umožní.

References

[1] J.R. Quinlan, “C4.5:Programs for Machine learning”, *Morgan Kaufmann Publishers*, 1993.

[2] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, “Classification and regression trees”, *Belmont CA: Wadsworth*, 1984.

[3] J.R. Quinlan, “Boosting, bagging and C4.5”, *Proceedings of AAAI’96.*, 1996.

[4] L. Breiman, “Random forests”, *Machine Learning*, vol. 45, p. 5–32, 2001.

[5] R.E. Shapire, Y. Singer, “Improved boosting algorithms using confidence-rated predictions”, *Machine Learning*, vol. 37, p. 297–336, 1999.

[6] H. Kim, W. Loh, “CRUISE User manual”, *Technical report 989, University of Wisconsin, Madison*, 2000.

[7] H. Kim, W. Loh, “Classification trees with unbiased multiway splits”, *Journal of the American Statistical Association*, vol. 96, p. 589-604, 2001

[8] W.-Y. Loh, Y.-S. Shih, “Split selection methods for classification trees”, *Statistica Sinica*, vol. 7, p. 815-840, 1997

[9] R.K. Bock, A. Chilingarian, M. Gaug, F. Hakl, T. Hengstebeck, M. Jiřina, J. Klaschka, E. Kotrč, P. Savický, S. Towers, A. Vaiciulis, W. Wittek, “Methods for multidimensional event classification: a case study using images from a Cherenkov gamma ray telescope”, *Technical report V-887, ICS AS CR*, 2003

[10] E.Kotrč “Analýza dat z projektu MAGIC-teleskop pomocí rozhodovacích stromů a lesů”, *Doktorandský den 02, Ústav informatiky AV ČR*, 2002

Practical model of serial computation

doktorand:
PAVEL KRUŠINA
Pod Vodárenskou věží 2, Praha

krusina@cs.cas.cz

školitel:
ROMAN NERUDA
Pod Vodárenskou věží 2, Praha

roman@cs.cas.cz

obor studia:
I-1

Abstract

In this article we design a new model of serial computation: the 86-RAM. It is a variant of a RAM model with a programming language based on an existing machine assembler code. This allows for exploiting tools as compilers and profilers for creating and measuring of the 86-RAM models. We show the polynomial equivalence between the RAM and 86-RAM models. Finally, we present an example of automatic model creation and measuring of its properties.

1. Introduction

There is a gap between complexity theory and a programming practice. While the theoretical RAM model of serial computation is clear and widely accepted, the existing program developing tools target mainly real computers and their machine codes.

In this article we aim for blurring the edge between theoretical models of serial computation and objects they are to model – the computational processes of real computers. We do it by enhancing the RAM model definition such that its programming language is a real computer assembler. This approach has the advantage of generating such models programs by machine compilation from high-level programming languages. Also the possibility of direct execution and measurements of these models is valuable especially for the complex computations. The work presented in this text is a part of a larger project geared at both the realistic parallel model and the computer implementation of execution and measurement environment for distributed computations.

We start with the RAM model definition and the definition of the x86 assembler language, we compare them and notice their similarities and differences. Then we design a new RAM-like model with the x86 instruction set and discuss its properties. Next, we present the equivalence theorems for the traditional RAM and the new 86-RAM models. Finally, we do an experiment of executing and measuring the 86-RAM program on a computer.

LOAD	operand
STORE	operand
ADD	operand
SUB	operand
JUMP	label
JZERO	label
READ	operand
HALT	

Table 1: The RAM instruction set.

2. Existing models

Here we present an overview of the two building blocks we will lately use to design our new model: the RAM sequential machine model and the assembler language.

Since we aim for parallel models design in near future, we use an uniprocessor version of PRAM as a RAM model definition. This ensures that there are no minor non-consistencies between the RAM and PRAM models [3].

Definition 1 *A* random access machine *(RAM) consists of a single processor P, an unbounded memory, a set of input registers, and a finite program. The processor has an accumulator and a program counter. All memory locations and accumulator are capable of holding arbitrary nonnegative integers. The program consists of a sequence of possibly labeled instructions chosen from the list above (Table 1). A program is nondeterministic if some label occurs more than once, deterministic otherwise.*

Each operand may be a literal, an address, or an indirect address.

Initially the input to the RAM is placed in the input registers, one bit per register, all memory is cleared, the length of the input is placed in the accumulator, and the processor is started. At each step in the computation the processor executes the instruction given by its program counter in one unit of time, then advances its counter by one unless the instruction causes a jump.

A READ instruction uses the value of the operand to specify one of the input registers; the contents of the selected register is placed in the accumulator. A HALT instruction causes the processor to stop running.

Execution continues until the processor executes HALT instruction. The input is accepted only if there is some computation in which the processor halts with one in its accumulator; the time required to accept the input is the minimum over all such computations of the number of instructions executed by the processor.

Definition 2 *A language L is in the class deterministic (nondeterministic) $T(n)$*-time-RAM *if there is a deterministic (nondeterministic) RAM M such that for all words x of length n, x is in L if and only if x is accepted by M and requires time at most $T(n)$.*

Next we remind a von Neumann machine definition.

Definition 3 *A* von Neumann machine *consists of a single processor P, a bounded memory, and a finite program. The processor has an accumulator, and a program counter, and possibly other registers. All memory locations and accumulator are capable of holding bounded integers. The program consists of a sequence of addressable instructions chosen from the fixed list. A program is always deterministic.*

Operands may be literals, addresses, or indirect addresses depending on the actual instruction.

Initially the input to the von Neumann machine is placed in the memory, the processor is started. At each step in the computation the processor executes the instruction given by its program counter in one unit of time, then advances its counter by one (even if the instruction is a jump).

Execution continues until the processor executes program halting instructions. The input is accepted only if the computation executes acceptance signaling instructions. Since the machine is deterministic, the acceptance time is simply the time of program run.

Definition 4 *A language L is in the class deterministic $T(n)$*-time-vN *if there is a von Neumann machine M such that for all words x of length n, x is in L if and only if x is accepted by M and requires time at most $T(n)$.*

Assembler language is a human readable form of von Neumann machine code. Since there are more real implementations of von Neumann machine, there are also more variants of the assembler language. In the next part of this article we focus on the Intel Architecture 32 (IA32) machines and its assembler language called the x86 assembler. For a complete definition of the x86 assembler language, the instruction semantic and the machine description one can read [2].

To illustrate the similarities and differences between low level languages like assembler and RAM on one hand, and high level languages like C++ on the other, we have programmed a simple subroutine of adding two integer vectors together (Table 2). The stack frame prologue and epilogue codes for assembler version is removed since it has no counterpart in RAM. According to this the function header should be removed also from the C++ version but it is left in place to help reader to understand the subroutine interface.

3. Bridging model

In this section we want to merge the two approaches discussed before and come with a new hybrid model combining the advantages of both.

Let us start with a list of important features not shared by the two models. The x86 programs are deterministic – this is common to all real implementations of von Neumann machine. The x86 has only bounded memory and stores only bounded integer numbers – these trivially come from the physical limitations common to all real computers. It is not so strong limitation as it may seem. Because we only compare characteristics of running programs, and a program exceeding the hard limits of a given real computer cannot continue, the programs we take into account always have enough memory. On the other hand, even on RAM, the accepting program (ending in the accepting state in a finite time) can only use a finite number of memory. As opposed to RAM, the x86 instructions do not always take the same time, they reflect the actual operation complexity and last proportionally to it.

There are no compilers from commonly used high-level programming languages into RAM as well as there are no profiling or performance control tools for RAM.

While the clean and simple design of theoretical RAM model is surely its advantage, the lack of modern programming tools negates this in a considerable degree. We can think of programming only small scale problems in RAM simply because of limitations of human programmers. Programming in such a low level language as RAM or assembler is quite expensive in term of human effort and work. For larger problems it is much more complicated to write them or debug them in the low level language directly than to write and maintain them in some suitable high level language like C or C++ which is translated to the low level language by the automatic compiler.

For example our generalized genetic algorithm core module [5] has about 1000 lines of extended C++ code while its x86 assembler version takes about 182000 lines. Since the x86 language has a quite large instruction set, the same code translated into RAM could have even more lines.

```
C++ function:                                   variables mapping:
------------                                    -----------------

void AddVec(int*a,int*b,int n)                         asm      RAM
{
    for(int i=0; i<n; i++) a[i]+=b[i];          n      ebx      [3]
}                                               i      edx
                                                a      ecx      [1]
                                                b      esi      [2]


 x86 assembler (in AT&T syntax):                RAM machine:
 ------------------------------                 -----------


        xorl      %edx, %edx                    .L6:
.L6:                                                 LOAD       *1
        movl      (%esi,%edx,4), %eax                ADD        *2
        addl      %eax, (%ecx,%edx,4)                STORE      *1
        incl      %edx                               LOAD       [1]
        cmpl      %ebx, %edx                         ADD         1
        jl        .L6                                STORE      [1]
.L8:                                                 LOAD       [2]
        ret                                          ADD         1
                                                     STORE      [2]
                                                     LOAD       [3]
                                                     SUB         1
                                                     JZERO      .L8
                                                     STORE      [3]
                                                     JUMP       .L6
                                                .L8:
                                                     HALT
```

Table 2: The RAM and assembler languages fragments.

Definition 5 *A* x86 random access machine *(86-RAM) consists of a single processor P, an unbounded data segment, an input segment , and a finite program. The processor has a fixed set of registers including a program counter. All memory locations and registers are capable of holding arbitrary nonnegative integers. The program consists of a sequence of addressable IA-32 instructions [2]. Operands may be literals, addresses, or indirect addresses according to the specification. Initially the input to the RAM is placed in the input segment and pointer to it is passed to the 86-RAM machine in the* `ebp` *register. The whole data segment is cleared, the length of the input is placed in the* `eax` *register, and the processor is started at the beginning of the program. At each step in the computation the processor executes the instruction given by its program counter in one unit of time, then advances its counter by one. A* `ret` *instruction of the top level program routine causes the processor to stop running. Execution continues until the processor executes top level* `ret` *instruction. The input is accepted only if the computation ends with non-zero in the* `eax` *register.*

Definition 6 *A language L is in the class $T(n)$*-time-86-RAM *if there is a 86-RAM M such that for all words x of length n, x is in L if and only if x is accepted by M and requires time at most $T(n)$.*

Considering the properties of the new model, we claim the 86-RAM model has these advantages: There are freely available compilers from several modern high level languages into x86 assembler and so to the 86-RAM programs. The 86-RAM programs are directly executable on a large set of IA32 computers if the execution happen to fit into memory and number size bounds given by the computer. There are freely available tools for measuring performance characteristics of a running program applicable also to running 86-RAM programs. The 86-RAM model is close enough to traditional deterministic RAM model, so we can compare and share results on both these architectures.

4. Equivalence of (deterministic) RAM and 86-RAM models

In this section we present a discussion of the RAM and 86-RAM models polynomial equivalence. For simulation of a RAM machine on a 86-RAM the complete proof is shown. For the opposite direction the program translation function could be constructed similarly, but because of a much richer IA-32 instruction set, it is not completely included.

Theorem 1 *For any deterministic RAM machine M running in time T, there is an equivalent 86-RAM machine N bounded by time Tc, where the c is a program independent constant.*

Proof. We construct the machine N by direct translation of the machine M program. We start at the beginning of the M program and according to Table 3 translate each RAM instruction into the 86-RAM language. All labels are kept unchanged at their places. The result is a correct and equivalent 86-RAM program. □

Let us note two observations: first, if the 86-RAM machine N is generated this way, the `ebp` register is never modified and so it is safe to use it as input segment base during the whole computation. Second, since in such 86-RAM machine is no call of subroutines, every `ret` instruction in it is the top level `ret` and thus equivalent to HALT instruction of RAM.

Theorem 2 *For any 86-RAM machine N running in time T, there is an equivalent RAM machine M bounded by time $Tp(I)$, where $p(I)$ is a program independent polynomial on the largest used number size hold in a single location.*

Sketch of proof. As in the previous case, we construct the machine M by direct translation of the machine N program. But there are several issues we should take care of: First, the RAM machine M should simulate 86-RAM features like larger set of registers and special memory segment of stack in its only memory space (see Figure 1). Second, the 86-RAM instruction set includes codes for which the emulation on RAM takes

RAM instruction		86-RAM instructions (in AT&T syntax)	
LOAD	const	movl	$const,%eax
LOAD	[memloc]	movl	memloc,%eax
LOAD	*indirect	movl movl	indirect,%ebx (%ebx),%eax
STORE	[memloc]	movl	%eax, memloc
STORE	*indirect	movl movl	indirect,%ebx %eax, (%ebx)
ADD	const	addl	$const, %eax
ADD	[memloc]	addl	memloc, %eax
ADD	*indirect	movl addl	indirect, %ebx (%ebx), %eax
SUB	const	subl	$const, %eax
SUB	[memloc]	subl	memloc, %eax
SUB	*indirect	movl subl	indirect, %ebx (%ebx), %eax
JUMP	label	jmp	label
JZERO	label	cmpl je	$0,%eax label
READ	idx	movl movl	idx, %ebx (%ebp,%ebx),%eax
HALT		ret	

Table 3: The RAM to 86-RAM translation table.

Figure 1: The RAM simulating 86-RAM.

Operator name	# of calls =n	Float ops. =f	D(f)	All ops. =a	D(a)	Cycles =c	D(c)
RND	62	121	1	9922	57	34097	3896
Mutation	276	15	0	3929	3	29008	4944
Average	111	60	0	3873	3	21261	1313
3-xover	19	240	0	12141	3	39882	1035
Euclidean	988	91	0	3241	3	20643	2340
Manhattan	809	90	0	3236	2	18667	893

Table 4: The 86-RAM execution measurements.

more than constant time like `mull` and `divl`, but all are bounded by a polynomial on operand size. The 86-RAM instruction set is large and the complete translation table from 86-RAM to RAM is beyond limits of this text. □

5. Example

Now we can proceed to exploiting the new model advantages. Imagine we want to model a complex computation such as genetic algorithm (GA). We have chosen to implement the algorithm in C++, however to obtain the 86-RAM model program text, we only have to generate it with the C++ compiler. When the programming and debugging parts are over, we can run the code, and by accessing the computer hardware counters perform measurings of the running code which are directly applicable to the model as well. This way we can learn how many instruction does it take to perform any given procedure or how much time does it cost. From such a data we can estimate the costs of runs not actually performed.

Here we present performance measurements results of a genetic operator package module (Figure 2). The code was written in extended C++, compiled into x86 machine code and measured during GA evolution. It shows the key ideas of our approach while it is still simple enough.

The examined code is a part of AOPack30 module responsible for performing creation, genetic, and metric operations on vectors of 30 double precision floating point numbers. The program was profiled using the PAPI library [4] and the results are listed in Table 4. Each line stands for a single operator. The first (RND) is creation operator, the next three (Mutation, Average, and 3-xover) are representants of genetic operators, and the last two (Euclidean and Manhattan) are metric operators. For each operator the total number of invocations during the testing n, the average number of floating point operations in one invocation f, the average number of all operations a, the average number of processor cycles per invocation c, together with variances D of these are counted. The processor cycles are connected through the CPU frequency to the real time used on the computation, so they allow us to estimate the run times on different computers as well.

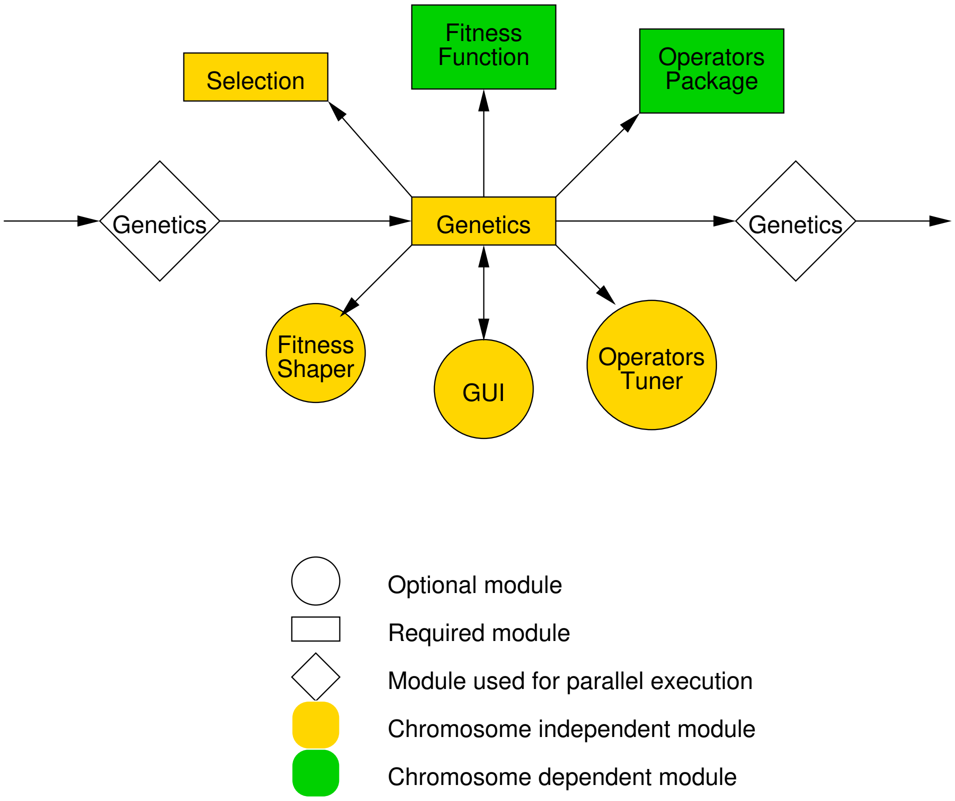


Figure 2: The GA decomposition.

6. Conclusion

In this article we have discussed differences and similarities of real computer assembler languages and theoretical models of computation like RAM. We observed that the differences were not fundamental, we could design RAM-like model based on an assembler language, and we could simulate RAM and 86-RAM models on each other effectively (Theorems 1 and 2). We have also demonstrated that using the assembler-like model allows us to write programs in modern high-level languages and then to use automatic compilers to generate the model. Further we can use existing performance tools to do measurements on the running program which are directly applicable for the model. Currently we are developing a parallel and asynchronous model upon the 86-RAM suitable for modeling complex distributed multi-agent systems like parallel genetic algorithm etc.

References

[1] Borland. *Turbo Assembler Quick Reference Guide*. Borland International, 1991.

[2] Intel Corporation. IA-32 Intel architecture software developer's manual volume 2: Instruction set reference, ftp://download.intel.com/design/pen-tium4/manuals/24547112.pdf.

[3] S. Fortune and J. Wylie. Parallelism in random access machines. In *Proceedings of the 10th Symposium on theory of Computing*, pages 114–118, 1978.

[4] Performance Application Programming Interface. http://icl.cs.utk.edu/projects/papi/.

[5] Pavel Krušina. AGenetics manual page, http://www.cs.cas.cz/bang/bang3/doc/www/man /agenet-ics.html.

Learning of Generalized Regularization Networks

doktorand:
PETRA KUDOVÁ
Institute of Computer Science,
Academy of Sciences of the Czech Republic,
Prague, Czech Republic
petra@cs.cas.cz

školitel:
ROMAN NERUDA
Institute of Computer Science,
Academy of Sciences of the Czech Republic,
Prague, Czech Republic
roman@cs.cas.cz

obor studia:
I1 Teoretická informatika

Abstract

In this paper we discuss regularization techniques and show that from regularization principles one can derive approximation schemes that are equivalent to feedforward neural networks with one hidden layer, which we will refer to as *regularization networks*. Several learning algorithms for regularization networks are discussed.

1. Introduction

In general a feedforward neural network can be viewed as a black-box with several inputs and outputs. The network is able to modify its parameters so as to fit the desired behaviour described by a set of examples, that is a set containing possible inputs together with corresponding outputs. Such a system, with the ability to learn autonomously a given task, can be very useful in many real life applications, namely in prediction, classification, control, etc. This learning is known as *supervised learning* or *learning from examples*. It turns out, that there is a close connection between the function approximation problem and the supervised learning of neural networks.

While training a neural network, we are given a set of input/output pairs $\{\vec{x}_i, \vec{y}_i\}_{i=1}^m$ (called *training set*) and we are looking for the function that represents the relation between the inputs $\vec{x}_i$ and the outputs $\vec{y}_i$ the best. A function in some a priori given form is considered, corresponding to the type and architecture of the network as well as some other "reasonable" assumption about the approximation function (such as smoothness).

In the section 2 we will summarize the regularization techniques and show how different types of regularization schemes, i.e. different regularization networks, can be derived. In the section 3 we describe several algorithms that can be used for learning of regularization networks.

2. The regularization approach

Given a set $\{(x_i, y_i) \in R^d \times R\}_{i=1}^N$ of data that was obtained by random sampling of some function f (belonging to some space of functions X defined on R^d), generally in the presence of noise, our goal is to recover the function f from the data, or find the best estimate of it. It can be done by *Empirical Risk Minimization*, i.e. finding a function that minimizes the functional $H[f] = \sum_{i=1}^N (f(\vec{x}_i) - y_i)^2$ over the *hypothesis space* $\mathcal{H}$, that is a chosen function space. Since this problem is generally ill-posed, we add an additional member called *stabilizer* to $H[f]$. The stabilizer typically forces the *smoothness* of the minimizing function, in the sense that two similar inputs correspond to two similar outputs. We get:

$$H[f] = \sum_{i=1}^{N} (f(\vec{x}_i) - y_i)^2 + \gamma \Phi[f], \tag{1}$$

where $\Phi[f]$ is a stabilizer and $\gamma > 0$ is a *regularization parameter* that controls the trade-off between the smoothness and closeness to data.

Tikhonov [8] proposed to choose a symmetric, positive-definite kernel function K and defined the stabilizer by means of the norm $||\cdot||_K$ in $\mathcal{H}_K$, that is the Reproducing Kernel Hilbert Space (RKHS) determined by the kernel K. So, we minimize the functional:

$$H[f] = \frac{1}{m} \sum_{i=1}^{m} (y_i - f(\vec{x}_i))^2 + \gamma ||f||_K^2, \tag{2}$$

over the hypothesis space $\mathcal{H}_K$. If we take the functional derivative with respect to f, apply it to $\bar{f} \in \mathcal{H}$, and set equal to 0:

$$\frac{1}{m} \sum_{i=1}^{m} (y_i - f(\vec{x}_i)) \bar{f}(\vec{x}_i) - \gamma \langle f, \bar{f} \rangle = 0 \tag{3}$$

where $\langle \cdot, \cdot \rangle$ is an inner product in RKHS. By setting $\bar{f} = K_{\vec{x}} = K(\cdot, \vec{x})$ we get

$$f(\vec{x}) = \sum_{i=1}^{m} c_i K_{\vec{x}_i}(\vec{x}) \qquad c_i = \frac{y_i - f(\vec{x}_i)}{m\gamma}. \tag{4}$$

Another form of smoothness functional based on Fourier transform was proposed by **Poggio and Girrosi** in [2]:

$$\Phi[f] = \int_{R^d} d\vec{s} \frac{|\tilde{f}(\vec{s})|^2}{\tilde{G}(\vec{s})}, \tag{5}$$

where $\tilde{f}$ indicates the Fourier transform of f , $\tilde{G}$ is some positive function that goes to zero for $||s|| \to \infty$ (i.e. $1/\tilde{G}$ is a high-pass filter).

Under slight assumptions on $\tilde{G}$ (as to be symmetric, so that its Fourier transform G is real and symmetric) is possible to show that the solution minimizing the functional (1) has the form:

$$f(x) = \sum_{i=1}^{m} c_i G(\vec{x} - \vec{x}_i) + \sum_{\alpha=1}^{k} d_\alpha \psi_\alpha(\vec{x}), \tag{6}$$

where $\{\psi_\alpha\}_{\alpha=1}^k$ is a basis in the k-dimensional space $\mathcal{N}$ of the functional Φ (in most cases a set of polynomials). Coefficients d_α and c_i depend on the data and satisfy the following linear system:

$$(G + \gamma I)\vec{c} + \Psi^T \vec{d} = \vec{y} \tag{7}$$

$$\Phi \vec{c} = 0 \tag{8}$$

where I is the identity matrix, and we defined:

$$\vec{y} = (y_1, \ldots, y_m),\ \vec{c} = (c_1, \ldots, c_m),\ \vec{d} = (d_1, \ldots, d_k) \tag{9}$$

$$(G)_{ij} = G(\vec{x}_i - \vec{x}_j),\ (\Psi)_{\alpha i} = \psi_\alpha(\vec{x}_i) \tag{10}$$

The existence of the solution to the linear system above is guaranteed by the existence of the solution of minimization (1).

The real form of the stabilizer (5) depends on the choice of $\tilde{G}$:

- **Radial stabilizers** – those are radially symmetric ones, i.e. satisfying $\Phi[f(\vec{x})] = \Phi[f(R\vec{x})]$, where R is an arbitrary rotation matrix. The radial symmetry reflects the assumption, that all the input variables have equal relevance, that is, there are no privileged directions. They lead to a *radial basis function* $G(||x||)$ and the approximation model corresponds to the RBF networks [3].

  The important example is the *Gaussian function* $\tilde{G}(\vec{s}) = e^{-\frac{||\vec{s}||^2}{\beta}}$ resulting in the basis function $G(\vec{x}) = e^{-\frac{||\vec{x}||^2}{\beta}}$, where β is a positive parameter. As the Gaussian function is a positive definite function, $\Phi[f]$ is a norm, its null space contains only the zero element, and therefore the additional terms of equation (6) are not needed.

- **Tensor product stabilizers** – an alternative of choice for $\tilde{G}(\vec{s})$ in the form $\tilde{G}(\vec{s}) = \Pi_{j=1}^{d}\tilde{g}(s_j)$, where $\tilde{g}$ is an one-dimensional function. This leads to a tensor product basis function $G(\vec{x}) = \Pi_{j=1}^{d} g(x_j)$, where g is the Fourier transform of $\tilde{g}$. For positive definite functions g the functional $\Phi[f]$ is a norm and its null space is empty.

  An interesting example is the choice $\tilde{g}(s) = \frac{1}{1+s^2}$, which leads to the basis function:

  $$G(\vec{x}) = \Pi_{j=1}^{d} e^{-|x_j|} = e^{-\sum_{j=1}^{d}|x_j|} = e^{-||\vec{x}||_{L1}}. \tag{11}$$

  This basis function is interesting from the point of view of hardware implementation, since it requires only the computation of L_1 norm (instead the usual Euclidean norm L_2).

- **Additive stabilizers** – it is also possible to derive the class of *additive approximation schemes*, i.e. the schemes of the form $f(\vec{x}) = \sum_{\mu=1}^{d} f_\mu(x^\mu)$, where f_μ are one-dimensional functions. It can be done by the use of the stabilizer

  $$\Phi[f] = \sum_{\mu=1}^{d} \frac{1}{\theta_\mu}\int_R ds \frac{|\tilde{f}_\mu(s)|^2}{\tilde{g}(s)}, \tag{12}$$

  where $\theta_\mu > 0$ are parameters allowing us to impose different degrees of smoothness on the different additive components. The minimum of this functional except the null space terms has again the form $f(\vec{x}) = \sum_{i=1}^{N} c_i G(\vec{x} - \vec{x}_i)$, where $G(\vec{x} - \vec{x}_i) = \sum_{\mu=1}^{d} \theta_\mu g(x^\mu - x_i^\mu)$. The additive components are not independent, because θ_μ are fixed.

3. Learning algorithms

In the previous section we derived several approximation schemes, that are all in the form

$$f(\vec{x}) = \sum_{i=1}^{m} c_i G(\vec{x}_i, \vec{p}_i, \vec{x}), \tag{13}$$

where m is the number of data points, $\vec{x}_i$ are the given data samples, G is some basis function, p_i are possible additional parameters of basis function G and c_i are real coefficients, typically called *weights*. The scheme (13) is called *regularization network*. By the *generalized regularization network* we then understand the network, where the number of basis functions is lower than the number of data points.

Poggio, Girosi in [7] proposed the special type of the generalized regularization network:

$$f(\vec{x}) = \sum_{\alpha=1}^{n} c_\alpha G(W\vec{x} - W\vec{t}_\alpha) \qquad n < N, \tag{14}$$

where W is matrix defining the transformation of the input space, t_α are heuristically chosen centers.

Now we introduce several algorithms dealing with the learning of generalized regularization networks. We start with the algorithm for the regularization network proposed by Poggio and Smale in [7]:

Algoritmus 3.1: Poggio, Smale

1. Start with data $\{\vec{x}_i, y_i\}_{i=1}^m \subseteq X \times Y$.
2. Choose a symmetric, positive-definite function $K_{\vec{x}}(\vec{x}') = K(\vec{x}, \vec{x}')$, continuous on $X \times X$.
3. Create $f : X \to Y$ by

$$f(\vec{x}) = \sum_{i=1}^{m} c_i K_{\vec{x}_i}(\vec{x}) \tag{15}$$

and compute $\vec{c} = (c_1, \ldots, c_m)$ by solving

$$(m\gamma I + K)\vec{c} = \vec{y}, \tag{16}$$

where I is the identity matrix, K is the square positive-definite matrix $K_{i,j} = K(\vec{x_i}, \vec{x_j})$, and $\vec{y} = (y_1, \ldots, y_m)$, $\gamma > 0$ is real number.

The linear system of equations (16) in m variables is well-posed, since K is positive and $(m\gamma I + K)$ is strictly positive. The strength of this algorithm is in its simplicity, but on the other hand, in real tasks, the data set can be really huge and the algorithms of type of 3.1, where the number of basis functions is the same as the number of data points, are not feasible.

Therefore we are more interested in algorithms for generalized regularization networks. In our past work we have studied RBF neural networks, that are in fact a special type of the generalized regularization network. We used a network in the form:

$$f(\vec{x}) = \sum_{j=1}^{h} w_j \varphi\left(\frac{\| \vec{x} - \vec{c}_j \|_{W_j}}{b_j}\right), \tag{17}$$

where φ is some radial basis function, typically e^{-z}, h is the number of hidden units (basis functions). Note that in our model each hidden unit j has possibly different norm matrix W_j. In [6] we described three main approaches to the RBF network learning *Three step learning*, *Gradient learning* and *Genetic learning* and also demonstrated them on several experiments.

The lower number of basis functions in generalized regularization networks is outweighted by the following obstacles: the learning problem is no more well-posed and more network parameters (such as centers $\vec{c}_j$, widths b_j and matrices W_j) have to be determined. We are in a danger of *overfitting*, that means that our solution can go through all the data points (or most of them) and minimize the empirical risk $\sum_{i=1}^{N} (f(\vec{x}_i) - y_i)^2$ correctly, but oscillate too much and give a poor generalization results.

One way how to deal with this problem, is to add the regularization term to the error function similarly we did while minimizing the functional (1). (Note that now we are minimizing a function on the space of network parameters, while in section 2 and 5 we were minimizing a functional on some function space.)

Bishop in [1] proposed to measure the smoothness using the second derivatives of the approximating function:

$$E_{Bsh} = \sum_{i=1}^{m} (f(\vec{x}_i) - y_i)^2 + \gamma E_{reg}, \qquad E_{reg} = \sum_{i=1}^{m} \sum_{j=1}^{n} \left(\frac{\partial^2 f}{\partial x_j^2}(\vec{x})\right)^2, \tag{18}$$

where x_j is the j-th coordinate of $\vec{x}$.

Our *Three step learning* adopted to the error function (18) leads to the algorithm:

Algoritmus 3.2: Three step algorithm based on Bishop's regularization

1. Set the centers $\vec{c}_j$ as random samples from the data set or find suitable representatives by a vector quantization.

2. Find the values for b_j a $\Sigma_j^{-1} = W_j^T W_j$ by minimizing the error function

$$E(b_1, \cdots, b_h; \Sigma_1^{-1}, \cdots, \Sigma_h^{-1}) = \frac{1}{2} \sum_{r=1}^{h} \left[\sum_{s=1}^{h} e^{-\left(\frac{\| c_s - c_r \|_{W_r}}{b_r}\right)^2} \left(\frac{\| c_s - c_r \|_{W_r}}{b_r} \right)^2 - P \right]^2 , \tag{19}$$

   where P is an *overlap parameter.*

3. Find the values of w_j: Take derivative of (18) and put them equal to zero. Solve the linear system by pseudoinverse.

$$W = \Phi^+ D \ , \quad \Phi^+ = (\Phi^T \Phi)^{-1} \Phi \tag{20}$$

   where $D \in R^{h \times m}$, $\Phi \in R^{h \times h}$ and

$$D_{ij} = \sum_{t=1}^{k} d_j^{(t)} y_i(\vec{x}^{(t)}), \qquad y_j(\vec{x}^t) = e^{-\left(\frac{\| \vec{x}^t - c_j \|_{W_j}}{b_j}\right)^2} \tag{21}$$

$$\Phi_{qr} = \sum_{t=1}^{k} \left(y_q(\vec{x}^{(t)}) y_r(\vec{x}^{(t)}) + \lambda \sum_{i=1}^{n} \left(\frac{\partial^2 y_q^{(t)}}{\partial x_i^2} \frac{\partial^2 y_r^{(t)}}{\partial x_i^2} \right) \right) . \tag{22}$$

This algorithm is similar to Algorithm 3.1 and is also quite simple. The trickier parts are the step 1 and 2, since they are in fact based on heuristics. The solving of linear system, though ill-posed, can be successfully treated by known numerical methods. The main disadvantage of the algorithm, as well as in Algorithm 3.1, is the presence of parameter γ, that must be somehow estimated, and that may significantly influence the solution.

Our second approach – *Gradient learning* – is based on minimizing the error function by the gradient algorithm. In order to include regularization, we simply replace the error function by the function (18), compute its derivatives and apply the gradient descent algorithm. In order to avoid the parameter γ , we will use the error function without the regularization member, but we will control the generalization ability of the network during learning, similary to cross-validation.

Algoritmus 3.3: Gradient learning with the test for generalization

1. Split the data set DS to *training set* TS_1 and *testing set* TS_2 (for instance 90% for training set, 10% for testing set, depending on the number of data points).
2. $\forall j\ \vec{c}_j(i) \leftarrow$ random sample from TS_1
   $\forall j\ b_j(i), \Sigma_j^{-1}(i) \leftarrow$ small random value
   $i \leftarrow 0$
3. For all j and $p(i)$ in $\vec{c}_j(i), b_j(i), \Sigma_j^{-1}(i)$:
$$\Delta p(i) \leftarrow -\epsilon \frac{\delta E_1}{\delta p} + \alpha \Delta p(i-1), \qquad p(i) \leftarrow p(i) + \Delta p(i) \tag{23}$$
$$E_1 \leftarrow \sum_{\vec{x} \in TS_1} (f(\vec{x}) - y_i)^2 \qquad \text{(error on the training set)} \tag{24}$$
4. $i \leftarrow i + 1$
5. $E_2 \leftarrow \sum_{\vec{x} \in TS_2} (f(\vec{x}) - y_i)^2$ (error on the testing set)
6. If both E_1 and E_2 are decreasing, go to 3. If E_2 started to increase, STOP.

The last algorithm we want to introduce here is the *Genetic learning*. It is the application of stochastic optimization technique known as *Genetic algorithms* on the minimization of the error function. Since it requires no computation of derivatives of the error function, but only an evaluation of it, we can use any form of the error function.

Algoritmus 3.4: Genetic learning with the regularization

1. Create random population P_0 of N feasible solutions. $i \leftarrow 0$
2. For all $I \in P_i$ compute:
$$cost(I) = \sum_{i=1}^{m} (f_I(\vec{x}_i) - y_i)^2 + \gamma E_{reg} \tag{25}$$
   where f_I is the function represented by individual I.
3. If the lowest cost in the population is sufficient STOP.
4. Create empty P_{i+1} and repeat until P_{i+1} is full:

   Selection: select 2 individuals I_1, I_2 (lower cost $\leftrightarrow$ higher probability).

   Crossover: $I_1', I_2' \leftarrow$ crossover(I_1, I_2).

   Mutation: mutate(I_1'), mutate(I_2').

   Add I_1', I_2' into P_{i+1}.
5. $i \leftarrow i + 1$, goto 2.

The regulatization member E_{reg} in the step 2 is either the Bishop's one from the equation (18) or the one proposed by Tikhonov:

$$E_{reg} = \gamma \sum_{i=1}^{n} w_i^2 \tag{26}$$

More details can be found in [6] or [4].

4. Conclusion

We gave a summary of the use of regularization principles, showed how the regularization networks can be derived and how the different types of stabilizers lead to different types of regularization networks.

We proposed several algorithms for estimating the parameters of generalized regularization networks with radial basis functions. After slight adaptations, all of them can be used also for the other types of regularization networks from section 2.

We have tested these algorithms, without the regularization extension, for the network with the Gaussian basis function in our past work. The results of several experiments can be found in [6],[4] or [5]. Now we plan to tests our algorithms, including the regularization extension proposed in this paper, and other models described in section 2.

References

[1] C.M. Bishop. Improving the generalization properties of Radial Basis Function neural networks. 3:579–588, 1991.

[2] Tomaso Poggio Federico Girosi, Michael Jones. Regularization theory and Neural Networks architectures. *Neural Computation*, 7,No.2:219–269, 1995.

[3] Simon Haykin. *Neural Networks*. Macmillan College Publishing Company, New York, 1994.

[4] P. Kudová. Learning methods for RBF networks. Technical Report V-846, Institute of Computer Science, AS CR, Prague., 2001. p. 19-24.

[5] P. Kudová. Genetic and eugenic learning of RBF networks. Technical Report V-880, Institute of Computer Science, AS CR, Prague., 2002. p. 1-6.

[6] R. Neruda and P. Kudová. Hybrid learning of RBF networks. In P.M.A. Sloot, C.J.K. Tan, J.J. Dongarra, and A.G. Hoekstra, editors, *Computational Science*, pages 594–603. Berlin, Springer, 2002. Lecture Notes in Computer Science; 2331.

[7] Tomaso Poggio and Steve Smale. The mathematics of learning: Dealing with data. *Notices of the AMS*, 50, No.5:537–544, 2003.

[8] A.N. Tikhonov and V.Y. Arsenin. Solutions of ill-posed problems. Technical report, 1997.

Použití metody s lokálně omezeným krokem pro podmíněnou minimalizaci

doktorand:
MGR. CTIRAD MATONOHA
ÚI AVČR, Pod vodárenskou věží 2, Praha 8

školitel:
DOC. RNDR. JAN ZÍTKO, CSC.
MFF UK, Sokolovská 83, Praha 2

obor studia:
M6 - vědecko-technické výpočty

Abstrakt

V tomto příspěvku pojednáme o problému minimalizace nelineární funkce na množině omezení obsahující rovnosti i nerovnosti. Zavedením logaritmického barierového členu převedeme původní problém na ekvivalentní problém, ve kterém se vyskytují pouze omezení s rovnostmi a odvodíme soustavu lineárních rovnic pro jeho řešení, které je ekvivalentní úloze nalezení lokálně omezeného kroku kvadratické funkce s lineárním omezením. Sestrojíme iterační metodu, založenou na modifikované metodě s lokálně omezeným krokem, sloužící k jejímu řešení.

1. Metoda vnitřního bodu

Budeme se zabývat obecným problémem nalezení minima funkce f na množině dané omezeními ve tvaru rovností a nerovností

$$f(x) \;\to\; \min, \quad \text{vzhledem k} \quad c_I(x) \le 0, \quad c_E(x) = 0, \tag{1}$$

kde

$$f : \mathbb{R}^n \to \mathbb{R}, \quad c_I : \mathbb{R}^n \to \mathbb{R}^{m_I}, \quad c_E : \mathbb{R}^n \to \mathbb{R}^{m_E}$$

jsou dvakrát spojitě diferencovatelné funkce ($c_I \le 0$ je myšleno po složkách),

$$I = \{1, ..., m_I\}, \;\; E = \{m_I + 1, ..., m_I + m_E = m\}$$

a předpokládáme, že $m_E < n$.

Tento problém je obtížně řešitelný z důvodu výskytu nerovností $c_I(x) \le 0$. Abychom tyto nerovnosti odstranili, zavedeme vektor pomocných proměnných

$$s \equiv s_I = (s_1, ..., s_{m_I}) \in \mathbb{R}^{m_I}$$

a převedeme problém (1) na úlohu s rovnostmi a jednoduchými nerovnostmi

$$f(x) \;\rightarrow\; \min, \quad c_I(x) + s = 0, \quad s \geq 0, \;\; c_E(x) = 0. \tag{2}$$

Podstata metody vnitřního bodu spočívá v náhradě omezení $s \geq 0$ přidáním logaritmického barierového členu s konstantou $\mu > 0$ k funkci f. Tím dostaneme úlohu

$$F(x,s) = f(x) - \mu e^T \ln(\mathbf{S}_I)e \;\rightarrow\; \min, \quad h(x,s) \stackrel{\text{def}}{=} [c_I(x) + s,\; c_E(x)] = 0, \tag{3}$$

kde e je vektor se samými jedničkami a $\mathbf{S}_I = \mathrm{diag}(s_i,\; i \in I),$ která má pouze omezení ve tvaru rovností. Logaritmická barierová funkce vyžaduje, aby platilo $s_i > 0 \;\forall i \in I,$ což lze zajistit vhodným výběrem délky kroku. Pro $\mu \rightarrow 0$ dostaneme řešení původní úlohy (1).

Budeme používat označení

$$c(x) = [c_I(x), c_E(x)] = [c_1(x), ..., c_m(x)] \in \mathbb{R}^m,$$

$$h(x,s) = [h_I(x,s), h_E(x,s)] = [c_1(x) + s_1, ..., c_{m_I}(x) + s_{m_I}, c_{m_I+1}(x), ..., c_m(x)]^T \in \mathbb{R}^m,$$

$$[\mathbf{A}_I(x), \mathbf{A}_E(x)] = [\nabla_x h_I(x,s), \nabla_x h_E(x,s)] = [\nabla_x c_1(x), ..., \nabla_x c_m(x)] \in \mathbb{R}^{n \times m}.$$

Jestliže má Jacobiho matice $[\mathbf{A}_I(x), \mathbf{A}_E(x)]$ lineárně nezávislé sloupce, pak řešení $x_\star, s_\star$ problému (3) splňuje následující Karush-Kuhn-Tuckerovy podmínky (nutné podmínky pro extrém). Nechť

$$L(x,s,u) = F(x,s) + u^T h(x,s) = f(x) - \mu e^T \ln(\mathbf{S}_I)e + u^T h(x,s)$$

je Lagrangeova funkce problému (3) s multiplikátory $u = [u_I^T, u_E^T]^T \in \mathbb{R}^m$. Označme

$$\begin{aligned}
g_x(x,s,u) &= \nabla_x L(x,s,u) = \nabla_x f(x) + [\mathbf{A}_I(x), \mathbf{A}_E(x)]\, u \\
g_s(x,s,u) &= \nabla_s L(x,s,u) = -\mu \mathbf{S}_I^{-1} e + u_I = -\mu \mathbf{S}_I^{-1} e + \mathbf{U}_I e \\
\mathbf{G}_{xx}(x,s,u) &= \nabla_{xx}^2 L(x,s,u) = \nabla_{xx}^2 f(x) + \sum_{k=1}^{m} u_k \nabla_{xx}^2 c_k(x) \\
\mathbf{G}_{ss}(x,s,u) &= \nabla_{ss}^2 L(x,s,u) = \mu \mathbf{S}_I^{-2}
\end{aligned}$$

její gradienty a Hessovy matice. Pak existuje vektor $u_\star \in \mathbb{R}^m$, že platí

$$\nabla_x L(x_\star, s_\star, u_\star) = 0, \quad \nabla_s L(x_\star, s_\star, u_\star) = 0, \quad \nabla_u L(x_\star, s_\star, u_\star) = 0.$$

Hledáme tedy řešení $x_\star, s_\star, u_\star$, které splňuje

$$g_x(x,s,u) = 0, \quad g_s(x,s,u) = 0, \quad h(x,s) = 0. \tag{4}$$

2. Odvození soustavy rovnic

Základní metody pro řešení problému (3) jsou iterační a jejich iterační krok má tvar

$$x^+ = x + \alpha_x d_x, \quad s^+ = s + \alpha_s d_s, \quad u^+ = u + \alpha_u d_u,$$

kde $d_x \in \mathbb{R}^n$, $d_s \in \mathbb{R}^{m_I}$, $d_u \in \mathbb{R}^m$ jsou směrové vektory a $\alpha_x, \alpha_s, \alpha_u > 0$ jsou délky kroku. Pro nalezení směrových vektorů použijeme metodu odvozenou z Newtonovy metody aplikovanou na nelineární KKT systém (4), kde $\alpha_x = \alpha_s = \alpha_u = 1$.

Budeme uvažovat dva přístupy. Primární formulace vznikne použitím Newtonovy metody na soustavu (4). Jestliže nejprve vynásobíme druhou rovnici (4) maticí $\mathbf{S}_I$

$$g_s = -\mu \mathbf{S}_I^{-1} e + \mathbf{U}_I e = 0 \quad \Rightarrow \quad \mathbf{S}_I g_s = -\mu e + \mathbf{S}_I \mathbf{U}_I e = 0$$

a teprve na tuto výslednou rovnici aplikujeme Newtonovu metodu, dostaneme tzv. primárně-duální formulaci. Ta je výhodnější, vede na efektivnější algoritmy. Po aplikaci Newtonovy metody dostaneme tyto rovnice (vlevo primární, vpravo primárně-duální formulace a vynecháme proměnné x, s, u) - liší se pouze druhá

$$\begin{aligned} \mathbf{G}_{xx}d_x + [\mathbf{A}_I, \mathbf{A}_E]\, d_u &= -g_x \\ \mathbf{G}_{ss}d_s + [\mathbf{I}, 0]\, d_u &= -g_s \\ [\mathbf{A}_I, \mathbf{A}_E]^T\, d_x + [d_s^T, 0]^T &= -h \end{aligned} \qquad \begin{aligned} \mathbf{G}_{xx}d_x + [\mathbf{A}_I, \mathbf{A}_E]\, d_u &= -g_x \\ \mathbf{U}_I d_s + \mathbf{S}_I d_{u_I} &= -\mathbf{S}_I g_s \\ [\mathbf{A}_I, \mathbf{A}_E]^T\, d_x + [d_s^T, 0]^T &= -h \end{aligned}$$

Předpokládejme, že $d_u = \left[d_{u_I}^T, d_{u_E}^T\right]^T$ a nechť matice $\mathbf{B}_{xx}$ aproximuje Hessovu matici $\mathbf{G}_{xx}$, neboť v praxi místo druhých derivací počítáme jejich aproximace pomocí diferencí. Aby se vyškáloval vektor d_s, vynásobíme obě druhé rovnice diagonální maticí $\mathbf{D}_I \in \mathbb{R}^{m_I}$ a abychom dostali symetrickou soustavu, nahradíme neznámou d_s výrazem $\mathbf{D}_I^{-1} d_s$. Dostaneme tuto tzv. primární, resp. primárně-duální iterační metodu se soustavou lineárních rovnic pro neznámé $d_x, d_s, d_{u_I}, d_{u_E}$:

$$\begin{pmatrix} \mathbf{B}_{xx} & 0 & \mathbf{A}_I & \mathbf{A}_E \\ 0 & \bar{\mathbf{D}} & \mathbf{D}_I & 0 \\ \mathbf{A}_I^T & \mathbf{D}_I & 0 & 0 \\ \mathbf{A}_E^T & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} d_x \\ \mathbf{D}_I^{-1} d_s \\ d_{u_I} \\ d_{u_E} \end{pmatrix} = - \begin{pmatrix} g_x \\ \mathbf{D}_I g_s \\ h_I \\ h_E \end{pmatrix}$$

kde

$$\bar{\mathbf{D}} = \mu \mathbf{D}_I \mathbf{S}_I^{-2} \mathbf{D}_I, \quad \text{resp.} \quad \bar{\mathbf{D}} = \mathbf{D}_I \mathbf{S}_I^{-1} \mathbf{U}_I \mathbf{D}_I.$$

Jestliže pro primární, resp. primárně-duální formulaci položíme

$$\mathbf{D}_I = \frac{1}{\sqrt{\mu}} \mathbf{S}_I, \quad \text{resp.} \quad \mathbf{D}_I = (\mathbf{S}_I \mathbf{U}_I^{-1})^{\frac{1}{2}}$$

(v praxi lze volit i jiná vyjádření matice $\mathbf{D}_I$), platí $\bar{\mathbf{D}} = \mathbf{I}$ a výsledný systém má tento tvar

$$\begin{pmatrix} \mathbf{B} & \mathbf{A} \\ \mathbf{A}^T & 0 \end{pmatrix} \cdot \begin{pmatrix} d \\ d_u \end{pmatrix} = - \begin{pmatrix} g \\ h \end{pmatrix}, \tag{5}$$

kde

$$d \in \mathbb{R}^{n+m_I}, \ \ \mathbf{B} \in \mathbb{R}^{(n+m_I)\times(n+m_I)}, \ \ g \in \mathbb{R}^{n+m_I}, \ \ \mathbf{A} \in \mathbb{R}^{(n+m_I)\times(m_I+m_E)}, \ \ h \in \mathbb{R}^{m_I+m_E},$$

přičemž

$$d = \begin{pmatrix} d_x \\ \mathbf{D}_I^{-1} d_s \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_{xx} & 0 \\ 0 & \mathbf{I} \end{pmatrix}, \quad g = \begin{pmatrix} g_x \\ \mathbf{D}_I g_s \end{pmatrix} \tag{6}$$

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_I & \mathbf{A}_E \\ \mathbf{D}_I & 0 \end{pmatrix}, \quad h = \begin{pmatrix} h_I \\ h_E \end{pmatrix} \tag{7}$$

Systém (5) je dimenze $n + 2m_I + m_E$. Tuto velikost lze zmenšit částečnou eliminací. Z druhé rovnice (4) plyne

$$-\mu \mathbf{S}_I^{-1} e + \mathbf{U}_I e = 0 \quad \Rightarrow \quad \mathbf{S}_I \mathbf{U}_I e = \mu e$$

a jestliže $\mu \to 0$, pak pro libovolný index $i \in I$ platí buď $u_i \to 0$ nebo $s_i \to 0$. Množinu omezení s nerovnostmi rozdělíme na aktivní a neaktivní podmnožinu.

- Jestliže platí $s_i \le \varepsilon_I u_i,\ i \in I,$ nazveme příslušná omezení aktivní a označíme je symbolem $\hat{\ }$ spolu s příslušnými veličinami, tedy např. $\hat{c}_I(x), \hat{s}_I, \hat{h}_I, \hat{u}_I$. Jsou to ta omezení, pro která je $c_i(x),\ i \in I$, blízko nuly, přičemž $\hat{c}_I \in \mathbb{R}^{\hat{m}_I}$.

- Jestliže platí $s_i > \varepsilon_I u_i,\ i \in I,$ nazveme příslušná omezení neaktivní a označíme je symbolem $\check{\ }$ spolu s příslušnými veličinami, tedy např. $\check{c}_I(x), \check{s}_I, \check{h}_I, \check{u}_I$. Jsou to ta omezení, pro která je $u_i,\ i \in I$, blízko nuly, přičemž $\check{u}_I \in \mathbb{R}^{\check{m}_I}$, kde $\hat{m}_I + \check{m}_I = m_I$.

Ze soustavy (5) vyloučíme neaktivní omezení. Nejprve vyeliminujeme

$$\check{d}_s = -\left(\check{\mathbf{A}}_I^T d_x + \check{h}_I\right) \tag{8}$$

a dále po dosazení a úpravě dostaneme pro primární, resp. primárně-duální metodu

$$\check{d}_{u_I} = \mu\check{\mathbf{S}}_I^{-2}\left(\check{\mathbf{A}}_I^T d_x + \check{c}_I\right) + 2\mu\check{\mathbf{S}}_I^{-1}e - \check{\mathbf{U}}_I e, \quad \text{resp.} \quad \check{d}_{u_I} = \check{\mathbf{S}}_I^{-1}\check{\mathbf{U}}_I\left(\check{\mathbf{A}}_I^T d_x + \check{c}_I\right) + \mu\check{\mathbf{S}}_I^{-1}e. \tag{9}$$

Nakonec dosadíme do první rovnice za $\check{d}_{u_I}$. Po této eliminaci obsahuje systém (5) pouze aktivní omezení a má tento tvar

$$\begin{pmatrix} \bar{\mathbf{B}}_{xx} & 0 & \hat{\mathbf{A}}_I & \mathbf{A}_E \\ 0 & \mathbf{I} & \hat{\mathbf{D}}_I & 0 \\ \hat{\mathbf{A}}_I^T & \hat{\mathbf{D}}_I & 0 & 0 \\ \mathbf{A}_E^T & 0 & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} d_x \\ \hat{\mathbf{D}}_I^{-1}\hat{d}_s \\ \hat{d}_{u_I} \\ d_{u_E} \end{pmatrix} = -\begin{pmatrix} \bar{g}_x \\ \hat{\mathbf{D}}_I\hat{g}_s \\ \hat{h}_I \\ h_E \end{pmatrix} \tag{10}$$

kde pro primární metodu je

$$\bar{\mathbf{B}}_{xx} = \mathbf{B}_{xx} + \mu\check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-2}\check{\mathbf{A}}_I^T, \quad \bar{g}_x = g_x + \mu\check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-2}\check{c}_I + 2\mu\check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-1}e - \check{\mathbf{A}}_I\check{\mathbf{U}}_I e$$

a pro primárně-duální metodu

$$\bar{\mathbf{B}}_{xx} = \mathbf{B}_{xx} + \check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-1}\check{\mathbf{U}}_I\check{\mathbf{A}}_I^T, \quad \bar{g}_x = g_x + \check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-1}\check{\mathbf{U}}_I\check{c}_I + \mu\check{\mathbf{A}}_I\check{\mathbf{S}}_I^{-1}e.$$

Matice $\bar{\mathbf{B}}_{xx}$ a vektor $\bar{g}_x$ jsou omezené (předpokládáme, že původní matice $\mathbf{B}_{xx}$ a vektor g_x jsou omezené) a dimenze soustavy (10) je $n + 2\hat{m}_I + m_E$.

3. Výpočet směrových vektorů

Systém (10) lze řešit buď přímo užitím vhodného rozkladu nebo iteračně předpodmíněnou metodou Krylovových podprostorů pro symetrické indefinitní systémy. My však zvolíme jiný způsob, převedením na metodu s lokálně omezeným krokem. Bez újmy na obecnosti můžeme předpokládat, že jsou všechna omezení aktivní, tedy $\hat{m}_I = m_I$. Uvažujme problém minimalizace kvadratické funkce vzhledem k lineárnímu omezení:

$$Q(d) = \frac{1}{2}d^T\mathbf{B}d + g^T d \;\rightarrow\; \min, \quad \mathbf{A}^T d + h = 0, \tag{11}$$

kde $d, \mathbf{B}, g, \mathbf{A}, h$ mají tvar (6) a (7). Označíme-li $d_u = [d_{u_I}^T, d_{u_E}^T]^T \in \mathbb{R}^{m_I+m_E}$ Lagrangeův multiplikátor, lze ukázat, že problém (11) a problém nalezení směrových vektorů pro problém (3) jsou ekvivalentní, neboť vedou na stejnou soustavu rovnic, která má obecný tvar (5).

Původní problém nalezení (aktivních) směrových vektorů jsme tedy převedli na problém minimalizace kvadratické funkce, na který aplikujeme metodu s lokálně omezeným krokem. Po přidání omezení

$$\|d\| \le \Delta \tag{12}$$

nám vznikne vedle lineárního omezení $\mathbf{A}^T d + h = 0$ ještě další podmínka. Obě podmínky však mohou být nekompatibilní (norma řešení (11), kterou neznáme, může být větší než Δ). Z tohoto důvodu provedeme modifikaci metody s lokálně omezeným krokem zavedením tzv. vertikálního a horizontálního kroku. Budeme uvažovat řešení $d_\star$ ve tvaru $d_\star = d_V + d_H$.

Uvažujme nejprve tuto minimalizační úlohu

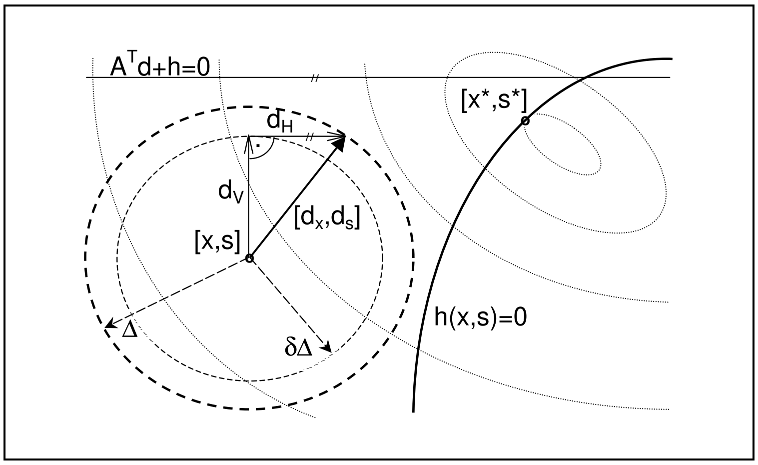
$$\|\mathbf{A}^T d + h\| \;\rightarrow\; \min, \quad \|d\| \le \delta\Delta \tag{13}$$

pro $0 < \delta < 1$ (např. $\delta = 0.8$). Řešení d_V této úlohy lze spočítat libovolnou metodou s lokálně omezeným krokem, např. metodou psí nohy.

Nyní přeformulujeme problém (11)-(12) takto:

$$Q(d) = \frac{1}{2}d^T\mathbf{B}d + g^T d \;\rightarrow\; \min, \quad \mathbf{A}^T d = \mathbf{A}^T d_V, \quad \|d\| \le \Delta. \tag{14}$$

Tato nová formulace obsahuje kompatibilní omezení, neboť volba $d = d_V$ splňuje obě podmínky. Řešení tohoto problému označíme d_H a spočítáme ho např. předpodmíněnou metodou sdružených gradientů. Na obrázku je vidět způsob výpočtu směrového vektoru pomocí vertikálního a horizontálního kroku.



4. Volba pokutové funkce

Metody s lokálně omezeným krokem vedou na krok $x^+ = x + \alpha_x d_x,\ s^+ = s + \alpha_s d_s$, kde buď $\alpha_x = 1$ a $\alpha_s \in (0, 1\rangle$ je takové, že $s^+ > 0$ (tzv. přijatelný krok) nebo $\alpha_x = \alpha_s = 0$ (tzv. nulový krok). Definujme

$$1_s = \begin{cases} 1, & \text{pokud } s + d_s > 0; \\ \min\{\alpha_s \in (0,1)\} \text{ takové, že } s + \alpha_s d_s > 0, & \text{pokud } s + d_s \le 0. \end{cases}$$

Označme $\alpha = [\alpha_x, \alpha_s]$ a pro $\alpha = 1$ definujme $1 = [1, 1_s]$. Uvažujme dále následující pokutovou funkci $P(\alpha)$ s koeficientem $\sigma > 0$:

$$P(\alpha) = F(x + \alpha_x d_x, s + \alpha_s d_s) + (u + d_u)^T h(x + \alpha_x d_x, s + \alpha_s d_s) + \frac{\sigma}{2} \|h(x + \alpha_x d_x, s + \alpha_s d_s)\|^2$$

a spočítejme skutečný a předpověděný pokles této funkce. Skutečný pokles je definován jako rozdíl $P(1) - P(0)$. Jestliže $Q_P(\alpha)$ je kvadratická aproximace funkce $P(\alpha)$, pak je rozdíl

$$Q_P(1) - Q_P(0) \approx P'(0) + \frac{1}{2} d^T \mathbf{B} d$$

předpověděný pokles funkce $P(\alpha)$. Abychom rozhodli, zda je krok přijatelný či nikoli, vytvoříme podíl skutečného a předpověděného poklesu. Jestliže

$$\frac{P(1) - P(0)}{Q_P(1) - Q_P(0)} > 0,$$

je krok přijatelný a poloměr Δ můžeme zvětšit. V opačném případě je krok nepřijatelný (nulový) a poloměr Δ je třeba snížit. Nutnou podmínkou pro aplikaci metody s lokálně omezeným krokem je však splnění nerovnosti $Q_P(1) - Q_P(0) < 0$. Ta je splněna, zvolíme-li

$$\sigma > -\frac{\frac{1}{2} d^T \mathbf{B} d + d^T g + d^T \mathbf{A} d_u}{d^T \mathbf{A} h}, \quad \text{kde} \quad d^T \mathbf{A} h < 0.$$

Parametr μ měníme v každé iteraci. Většina implementací metod vnitřního bodu volí hodnotu μ tak, že $0 < \mu < \frac{s^T u_I}{m_I}$, tedy $\mu = \frac{\lambda s^T u_I}{m_I}$ pro $\lambda \in (0, 1)$. V praxi se ukázalo, že algoritmus pracuje nejlépe tehdy, když jdou složky $s_i u_i$ stejnoměrně k nule. K tomuto účelu zavedeme veličinu

$$\varrho = \frac{\min_{i \in I}\{s_i u_i\}}{s^T u_I / m_I}$$

Zřejmě platí $0 < \varrho \leq 1$ a $\varrho = 1$ právě když je $s_i u_i$ konstantní $\forall i \in I$. Nyní použijeme následující heuristiku pro volbu λ a tedy μ, která se v praxi ukázala jako velmi efektivní:

$$\mu = \lambda \frac{s^T u_I}{m_I}, \quad \text{kde} \quad \lambda = 0.1 \cdot \min \left\{ \frac{1-\varrho}{20\varrho}, 2 \right\}^3$$

Poloměr Δ měníme v závislosti na hodnotě podílu $\frac{P(1)-P(0)}{Q_P(1)-Q_P(0)}$. Je-li blízko nuly, Δ zmenšíme, je-li blízko jedničky nebo větší než jedna, Δ zvětšíme.

Na závěr uvedeme algoritmus popsané metody.

Data: $f : \mathbb{R}^n \to \mathbb{R},\ c = [c_I, c_E] : \mathbb{R}^n \to \mathbb{R}^m$.
Zvolíme: $x \in \mathbb{R}^n,\ s \in \mathbb{R}^{m_I},\ u \in \mathbb{R}^m,\ \mu > 0,\ \varepsilon, \varepsilon_I \in (0,1),\ 0 < \beta < 1 < \gamma,\ 0 < \underline{\omega} < \overline{\omega} < 1$.

1. Určíme $\mathbf{A}_I(x),\ \mathbf{A}_E(x),\ g(x,s,u),\ h(x,s)$. Je-li $\mu < \varepsilon,\ \|g\| < \varepsilon,\ \|h\| < \varepsilon$, pak `STOP`.
2. Pomocí diferencí spočítáme druhé derivace funkcí $f(x)$ a $c(x)$, položíme $\Delta = \|g\|$ a určíme aktivní a neaktivní omezení $(s_i \leq \varepsilon_I u_i,$ resp. $s_i > \varepsilon_I u_i)$.
3. Metodou s lokálně omezeným krokem spočítáme vektory $d_x, \hat{d}_s, \hat{d}_{u_I}, d_{u_E}$, ze vzorců (8) a (9) vypočítáme vektory $\check{d}_s, \check{d}_{u_I}$, tím získáme d_s, d_u a položíme $d = [d_x^T, d_s^T]^T$.
4. Zvolíme $\sigma > 0$ tak, aby $\sigma > -\frac{\frac{1}{2} d^T \mathbf{B} d + d^T g + d^T \mathbf{A} d_u}{d^T \mathbf{A} h}$.
5. Spočítáme maximální délku kroku $\alpha_s \in (0,1\rangle$ takovou, aby platilo $s + \alpha_s d_s > 0$.
6. Vypočítáme podíl $\omega = \frac{P(1) - P(0)}{Q_P(1) - Q_P(0)}$. Je-li $\omega \leq 0$, zvolíme $\Delta < \|d\|$ a návrat na krok 3.
7. Spočítáme maximální délku kroku $\alpha_u \in (0,1\rangle$ takovou, aby platilo $u + \alpha_u d_u > 0$.
8. Položíme $x := x + d_x,\ s := s + \alpha_s d_s,\ u := u + \alpha_u d_u$.
9. Pokud $\omega < \underline{\omega}$, položíme $\Delta := \beta \|d\|$, pokud $\overline{\omega} < \omega$, položíme $\Delta := \gamma \Delta$. Dále spočítáme $\varrho = \frac{\min_{i \in I}\{s_i u_i\}}{s^T u_I / m_I}$, $\lambda = 0.1 \cdot \min\left\{\frac{1-\varrho}{20\varrho}, 2\right\}^3$, položíme $\mu = \lambda \frac{s^T u_I}{m_I}$ a návrat na krok 1.

5. Numerické experimenty

Výše uvedená metoda byla testována v prostředí UFO na třech množinách, každá z nich obsahuje 17 testovacích problémů s 1000 proměnnými. Výsledky jsou uvedeny v tabulce, kde jednotlivé sloupce znamenají

- NIT - celkový počet iterací
- NFV - celkový počet vyčíslení funkční hodnoty
- NFG - celkový počet vyčíslení gradientu
- NCG - celkový počet iterací metody sdružených gradientů
- NRS - celkový počet restartů
- TIME - celkový čas v sekundách
- NFAIL - celkový počet problémů z dané množiny, které se nepodařilo vyřešit

Množina	**NIT**	**NFV**	**NFG**	**NCG**	**NRS**	**TIME**	**NFAIL**
1	1106	1171	8522	26060	10	10.53	1
2	904	998	6185	10521	8	6.77	1
3	544	625	3989	7545	8	8.36	1

Informace o systému UFO a testovaných příkladech lze získat na adrese
http://www.cs.cas.cz/˜luksan/test.html.

References

[1] R.H.Byrd, J.C.Gilbert, J.Nocedal: *A Trust Region Method Based on Interior Point Techniques for Nonlinear Programming,* 1996

[2] R.H.Byrd, M.E.Hribar, J.Nocedal: *An Interior Point Algorithm for Large Scale Nonlinear Programming,* 1997

[3] A.R.Conn, N.I.M.Gould, P.L.Toint: *Trust-region methods,* SIAM 2000

[4] J.E.Dennis, L.N.Vicente: *On the convergence theory of trust-region-based algorithms for equality-constrained optimization,* SIAM J. Optimization, Vol.7, 1997, pp.927-950

[5] N.I.M.Gould, M.E.Hribar, J.Nocedal: *On the Solution of Equality Constrained Quadratic Programming Problems Arising in Optimization,* 2000

[6] L.Lukšan: *Metody s proměnnou metrikou,* Academia Praha, 1990

[7] L.Lukšan: *Numerické optimalizační metody pro úlohy bez omezujících podmínek,* Výzkumná zpráva č. V-640, 1995

[8] L.Lukšan, C.Matonoha, J.Vlček: *Interior point method for nonlinear nonconvex optimization,* to appear in Numerical Linear Algebra with Applications

[9] L.Lukšan, J.Vlček: *Numerical experience with iterative methods for equality constrained nonlinear programming problems,* Optimization Methods and Software, Vol.16, 2001

Současný stav dostupných softwareových systémů pro genetickou statistiku

doktorand:
MGR. MARKÉTA PAVLÍKOVÁ
EuroMISE Centrum, Ústav informatiky AV ČR, Pod Vodárenskou věží 2, Praha 8
pavlikova@euromise.cz

školitel:
JANA ZVÁROVÁ
EuroMISE Centrum, Ústav informatiky AV ČR, Pod Vodárenskou věží 2, Praha 8
zvarova@euromise.cz

obor studia:
Pravděpodobnost a matematická statistika

Abstrakt

Článek popisuje výsledky analýzy charakteristik dostupných softwareových systémů pro genetickou statistiku. Studie vznikla v rámci shromažďování informací o existujícím software při přípravě vlastního otevřeného modulárního systému pro analýzu genetických dat v EuroMISE Centru – Kardio.

1. Úvod

V současné době existují více než dvě stovky nejrůznějších softwareových systémů vhodných pro analýzu genetických dat. Jejich tvůrci pocházejí ze všech končin (internetového) světa, a stejně tak jsou po internetu roztroušeny stránky, kde je možné získat o software informace či možnost si ho stáhnout. I přes existenci několika málo seznamů, které jsou více či méně úplné a více či méně často aktualizované, může hledání vhodného software pro analýzu právě našich dat trvat i týden a ani pak nemusíme být úspěšní.

V minulém roce vznikla v rámci EuroMISE Centra – Kardio iniciativa za vznik vlastního otevřeného systému, který by nabídl výzkumníkům (zejména lékařům a biologům) jednoduchou možnost drobných analýz vlastních dat, aniž by museli vynakládat úsilí na učení se obecného software. Genetická data mají totiž svůj specifický charakter a aplikovat na jejich analýzu obecný software často vyžaduje vhled odborného statistika. Postupně se idea rozvinula do představy modulárního systému, který by byl intuitivní, otevřený a časem mohl pokrýt kromě jednoduchých aplikací také výsledky výzkumných aktivit Centra.

Součástí přípravy se také stal důkladný průzkum stávajícího publikovaného software a otevřených systémů pro analýzu genetických dat. Kromě potvrzení, že něco podobného naší koncepci není pro tuto chvíli zastoupeno, přinesla analýza řadu zajímavých faktů nejen o stavu, dostupnosti a složitosti genetického software, ale také o různých jazycích a platformách, pro které je software vytvářen.

2. Metody

Analýza byla založena na informacích shromážděných v pravděpodobně nejkvalitnějším (nejúplnějším) seznamu genetického software na `linkage.rockefeller.edu/soft` [1]. Ze seznamu byly extrahovány objektivní informace o jazyce, platformě a obsahu; subjektivně byla pak každá položka vyhodnocena podle dostupných informací co do velikosti a aktuálnosti. Velikost software se pohybovala na stupnici malý (jeden či několik málo spustitelných souborů, zabývá se jedním tématem/algoritmem), střední (několik metod/spustitelných souborů/témat) a komplexní (nahlíží data z několika úhlů, aplikuje různé metody a doplňky, stará se o data od kontroly zápisu po zobrazení). Aktuálnost byla hodnocena podle roku posledního update s hranicí posledních pět let a před rokem 1990 (kategorie nový/vylepšovaný; starší, ale udržovaný/používaný; starý, neudržovaný).

Obsah programu, tedy oblast genetické statistiky, kterou se zabývá, se určoval obtížně. Nakonec se kategorie ustálily na následujících: A (asociační studie, včetně TDT a case-control), L (parametrická i neparametrická vazebná analýza), G (hledání umístění genu v genetické mapě, analýza QTL), M (tvoření a práce s genetickými mapami), E (populační charakteristiky), D (tvorba a správa databáze), S (simulace), P (kreslení a správa rodokmenů), R (kreslení map), X (určování charakteristik rodokmenů včetně inference haplotypů), Y (výpočty velikosti souborů a síly testů), H (pomocné programy na konverzi dat mezi programy, kontrolu chyb apod.).

3. Výsledky

V databázi [1] bylo nalezeno celkem 240 různých programů pro práci s genetickými daty. Ve dvanácti případech byl program již dávno zahrnut do některého ze systémů a z analýzy byl vyřazen. Celkem tak do analýzy vstoupilo 228 položek, 12 z nich (5%) se zcela chybějícími hodnotami (nebylo možno získat žádné informace kromě názvu, výjimečně autora). Procenta jsou proto obvykle vztažena k počtu 216 položek s alespoň nějakými daty. Připomeňme, že procenta v tabulkách se ne vždy sčítají do 100%, protože některé položky vykazují více než jež jeden aspekt jak zaměření, tak třeba užitého programovacího jazyka nebo operačního systému.

3.1. Zaměření

Zaměření software v databázi (tabulka 1) se zdá odpovídat rozložení zájmů ve vědecké obci. Nejsilnější pozici zaujímá software pro vazebnou analýzu (26%), což je v souladu s tím, že vazebná analýza je typicky nejčastěji a historicky nejdéle prováděným typem analýz. Nepřekvapuje ani druhá pozice programů pro asociační studie (17%), které obvykle vazebnou analýzu doplňují a v poslední době si, zejména díky TDT a jeho variantám, získávají oblibu. Překvapením bylo vcelku malé množství programů pro hledání QTL (10%), které je vysvětlitelné existencí několika málo velmi dobrých a propracovaných algoritmů. Na rozdíl od vazebné analýzy, která používá velké množství metod, a jejichž velkou část není složité naprogramovat, hledání QTL a umisťování genů do mapy je poměrně obtížné a vedlo k rozvinutí mála, ale dobrých a rozšířeně používaných programů (Genehunter, MapMaker a jejich odvozeniny). Poměrně silně byl v databázi zastoupený software pro vytváření a zobrazování genetických map (celkem 15%) různé kvality a stáří, používaný obvykle pro data z experimentálního živočišného a rostlinného výzkumu.

Stanovováním populačních charakteristik, které kromě alelických frekvencí zahrnují i testování HWE a různé míry genetické vzdálenosti a rozmanitosti, se zabývá jen málo software (5%), a to navíc obvykle v rámci komplexního pohledu na genetická data. Důkladně a samostatně se této problematice věnuje pouze jeden nalezený program, švýcarský Arlequin.

Samostatnou kapitolu tvoří skupina pomocného software, který zahrnuje více než třetinu software v databázi (celkem 36%). Patří do něj jednak databázové systémy (8%), programy na výpočet velikosti souborů a síly testů (4%), simulační software (5%), velmi užitečný pro ověřování stabilnosti hypotéz a statistik, a programy na kontrolu a konverzi dat (12%). Zajímavostí je, že programy na konverzi datových formátů vznikaly především v dřívějších dobách, nyní jsou zpravidla, vzhledem k postupující standardizaci, automatickou součástí větších balíků.

kód	obor	počet zastoupených	procento zastoupených
A	asociační studie	37	17%
L	vazebná analýza	57	26%
G	umístění genu	22	10%
M	tvoření map	27	13%
R	kreslení map	6	3%
E	populační charakteristiky	10	5%
D	databáze	18	8%
S	simulace	11	5%
P	tvorba rodokmenů	21	10%
X	charakteristiky rodokmenů	18	8%
Y	pomocné výpočty	8	4%
H	pomocné programy	25	12%

Tabulka 1: Zastoupení jednotlivých oborů (n=216)

Poslední velkou skupinu z těchto pomocných programů tvoří software na kreslení rodokmenů (10%). V tomto případě se do databáze dostaly nejen programy doplňující analytický software, ale i komerční programy pro práci praktických lékařů a genetických poradců a dokonce i komerční kreslicí software pro kreslení rodinných rodokmenů soukromníků. Není bez zajímavosti, že tento je často předražený vzhledem k cenám licencí daleko schopnějšího software, nehledě k tomu, že i mimo [1] lze k tomuto účelu najít řadu open source programů. Téměř polovina kreslicího software v databázi byla placená (10 z 21), na druhou stranu tvořil tento většinu (přes 60%) placeného software (kterého bylo celkem 7%). Zbytek placeného software tvořily zejména databáze (často i s možností kreslení) a komplexní analytické celky (S.A.G.E., makra pro SAS, dvojice JoinMap a MapQTL apod.).

Zajímavý je rozdíl mezi software z oboru G (umisťování genu) a L+A (vazebná analýza a asociační studie) vzhledem k způsobu rozvíjení a vytváření nových systémů. V oboru G se časem vyprofilovaly dvě velké skupiny založené na základních algoritmech, Genehunter a MapMaker. Ostatní programy pak typicky základní systémy nějakým způsobem doplňují (rozšíření pro rodokmeny s komplikovanější strukturou, pro dva bialelické lokusy, pro sourozence, kreslicí doplněk) nebo zrychlují. Celkem tak software z těchto dvou skupin tvoří přes polovinu programů ve svém oboru. V oboru L+A je situace jiná, software tvoří rozmanitou směs, z níž vystupují dvě výraznější skupiny: Linkage s podobnou evolucí jako Genehunter (zobecňování, doplňky, případně zrychlení – Allegro), která se soustředí pouze na vazebnou analýzu, a Pangaea, která je spíše volným, ale jednotně spravovaným uskupením devíti programů různého zaměření od populačních charakteristik přes asociační studie až po vazebnou analýzu a kreslení rodokmenů. Tyto dvě rozdílné struktury dobře odrážejí kompaktní nebo naopak roztříštěný charakter daných oborů.

3.2. Rozsah a aktuálnost

Většina programů byla zařazena do kategorie malý (45%, viz tabulka 2). To vypovídá o tom, že značná část existujícího software jsou zčásti malé spustitelné soubory, které si výzkumníci píší, aby aplikovali právě jednu zkoumanou metodu, a zčásti pomocné utilitky, které se zabývají úpravou dat, zobrazováním a zkoušením. Přibližně třetina software se zařadila do kategorie střední (36%) a 26 programů (12%) bylo označeno jako komplexní. V této skupině se pochopitelně ocitly jednak úspěšné celky typu Linkage, Genehunter a MapMaker, jednak volná seskupení drobných utilit a programů (Pangaea, MKGST, linkage utility programs), samostatné propracované jednotky jako Merlin, Arlequin, Madeline, Mendel, a také komerční software jako makra pro SAS, S.A.G.E, Progeny a GAP. Souvislost komerčního software s velikostí není náhodná, placené programy tvoří čtvrtinu mezi všemi komplexními. Na druhou stranu je velmi příjemné zjištění, že ve všech oborech analýz existuje dobrý, rozvinutý a volně šiřitelný software, který si navíc dlouhodobým používáním získal jméno, jaké se komerčnímu software ve sféře genetické statistiky bude získávat velmi těžko.

velikost software	počet	procento
komplexní	26	12%
střední	77	36%
malý	97	45%
údaj nelze stanovit	16	7%
celkem	216	100%

stáří software	počet	procento
aktuální	111	51%
starší, ale udržovaný	95	44%
starý	9	4%
údaj nelze stanovit	1	1%
celkem	216	100%

Tabulka 2: Rozsah a aktuálnost jednotlivých softwareových systémů (n=216)

Z rozboru aktuálnosti software v databázi (tabulka 6) vyplynula potěšující informace, že polovina programů je nedávná nebo průběžně aktualizovaná. Uživatel si tedy u velké části může být jist, že software bude fungovat dobře (u aktualizovaných) nebo že na jeho připomínky k funkčnosti bude někdo reagovat a program vylepšovat. Z tabulky vztahu stáří a velikosti programu (tabulka 7) je pak patrný moderní příklon k větším celkům, který odpovídá postupnému vylepšování, sdružování a zobecňování genetického software.

		komplexní	střední	malý	celkem
starší a starý	počet	8	30	54	92
	řádkové %	9%	33%	59%	100%
aktuální	počet	18	47	43	108
	řádkové %	17%	44%	40%	100%
celkem	počet	26	77	97	200
	řádkové %	13%	39%	49%	100%

Tabulka 3: Vztah aktuálnosti a rozsahu jednotlivých softwareových systémů (n=216), p-hodnota χ^2-testu $0,0222$

3.3. Jazyky a platformy

Informace o jazycích a platformách byly sice zjišťovány objektivně, ale současně jsou nejméně spolehlivým záznamem v databázi. Jednak velmi často informace o jazyku chybí (u 39 položek, a nejen u komerčního software), jednak údaj o operačních systémech zastarává, jak přibývají nové verze, rozšířené pro další operační systémy. Někde bylo možno údaj dohledat na stránkách software, jinde se to nepodařilo. Přesto jsou výsledky poměrně vypovídající.

Většina software (59%, viz tabulka 4), zejména ten starší, byla psána pro operační systémy typu UNIX. Dost velká část programů pracuje na PC na platformách Windows (36%) a DOS (30%, společně pak 58%). Teprve v poslední době se objevují informace o verzích pro Linux (16%), což může být způsobeno jednak tím, že většina pracovišť je historicky vybavena stroji s některým UNIXem, a jednak tím, že velká část programů pro jiné platformy bude fungovat i pod Linuxem. Novinkou poslední doby jsou aplikace v Javě použitelné na libovolné platformě (2%). Celkem 18% programů bylo psáno i pod jiné platformy (MacOS, VMS).

operační systém	počet zastoupených	procento zastoupených
UNIX	111	59%
Linux	29	16%
DOS	56	30%
Windows	67	36%
jiný	33	18%

Tabulka 4: Přehled cílových operačních systémů (n=187)

Použitým programovacím jazykům vévodí C (40%, viz tabulka 5), následované vyrovnaně C++, Pascalem a Fortranem (shodně 19%). Některé programy požívají kombinace různých jazyků, např. Fortranu a C. Databázový software používá různé databázové jazyky (FoxPro, Paradox, dBase).

programovací jazyk	počet zastoupených	procento zastoupených
C	107	40%
C++	144	19%
Fortran	144	19%
Pascal	143	19%
Java	173	2%
jiný	139	21%

Tabulka 5: Použitých programovacích jazyků (n=177)

4. Závěr

Analýza seznamu statistického software na `linkage.rockefeller.edu/soft` potvrdila rozsah a pestrost, jakož i šíři záběru systémů pro analýzu genetických studií, a zaznamenala, že dost velká část programů je stále spravována, doplňována a vylepšována. Ukázal se patrný trend k vytváření větších funkčních celků, které budou provádět na jedněch datech všechny operace potřebné k analýze, a to buď v rámci jednoho systému, nebo v různých programech, ale se stejným datovým formátem (standardem se například stává formát Linkage).

Současně s těmito zjištěními se ukázalo, že představa a náplň nového otevřeného systému pro genetickou statistiku, tak jak vznikla v EuroMISE Centru, není dublována již existujícím software, a má smysl pokračovat v jeho realizaci. Zároveň je třeba vyzdvihnout fakt, že díky této studii vznikla databáze, ve které se bude možno orientovat o mnoho lépe než v textových stránkách [1], a vyhledání vhodného systému pro účely analýzy nebude muset být tolik úmornou prací.

References

[1] webová stránka `http://linkage.rockefeller.edu/soft`

[2] webové stránky jednotlivého software

Modelling of resonance characteristic of piezoelectric resonators

doktorand:
PETR RÁLEK

Katedra modelování procesů , FM TU v Liberci, Hálkova 6, Liberec 1

petr.ralek@vslib.cz

školitel:
ZDENĚK STRAKOŠ

Oddělení výpočetních metod, ÚI AV ČR, Pod vodárenskou věží 2, Praha 8

strakos@cs.cas.cz

obor studia:
Technická kybernetika

Abstract

In this paper, we describe the finite element (FEM) model of the piezoelectric resonator based on the physical description of the piezoelectric material. Discretization of the problem then leads to a large sparse linear algebraic system, which defines the generalized eigenvalue problem. Resonance frequencies are subsequently found by solving this algebraic problem. The results of the testing problem are introduced and other possibility for solving the eigenvalue problem, based on the connection of our problem with algebraic problem occuring in the theory of control, is proposed.

1. PROBLEM DESCRIPTION

1.1. Physical description

Piezoelectric resonator is the thin stick or wafer made of the piezoelectric material, with two or more electrodes on its surface (see, e.g., [2]). In consequence of harmonic electric loading, the resonator oscillates. The most important parameters, describing the behavior of the resonator, are its *resonance frequencies* (or *eigenfrequencies*) - frequencies of the oscillations with maximal amplitudes in some characteristic directions. A crystal made of piezoelectric material represents a structure in which the deformation and electric field depend on each other. A deformation (impaction) of the crystal induces electric charge on the crystal's surface. On the other hand, subjecting a crystal to electric field causes its deformation. This process is described by two state equations - the **generalized Hook's law** (1) and the **equation of the direct piezoelectric effect** (2), see, e.g.[2]

$$\mathrm{T_{ij}} = c_{ijkl} \cdot \mathrm{S_{kl}} + d_{ijk} \cdot \mathrm{E_k}, \qquad i, j = 1, 2, 3. \tag{1}$$

$$\mathrm{D}_k = d_{kij} \cdot \mathrm{S_{ij}} + \varepsilon_{kj} \cdot \mathrm{E_j}, \qquad k = 1, 2, 3, \tag{2}$$

where, as in all similar terms troughout the article, we will use the Einstein's additive rule (e.g. $a_{ij}b_j = \sum_{j=1}^{3} a_{ij}b_j$). The Hook's law (1) describes the dependence between the **stress tensor T**, the **strain tensor S** and the **vector of intensity of electric field E**,

$$\mathrm{S}_{ij} = \frac{1}{2}\left[\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right], \qquad i,j = 1,2,3,$$

$$\mathrm{E_k} = -\frac{\partial \tilde{\varphi}}{\partial x_k}, \qquad k = 1,2,3,$$

where $\tilde{\mathbf{u}} = (\tilde{u}_1, \tilde{u}_2, \tilde{u}_3)^{\mathrm{T}}$ is the **displacement vector** and $\tilde{\varphi}$ is the **electric potential**. The strain tensor **S** and the stress tensor **T** are symmetric [2]. The equation of the direct piezoelectric effect (2) describes the dependence between the **vector of electric flux density D**, the strain and the intensity of electric field. Quantities c_{ijkl}, d_{kij} and ε_{ij} represent symmetric material tensors,

$$c_{ijkl} = c_{jikl} = c_{ijlk} = c_{klij}, \quad d_{ijk} = d_{ikj} = d_{kij}, \quad \varepsilon_{ij} = \varepsilon_{ji}.$$

The state equations (1) and (2) represent a linear approximation of the thermodynamic state equations with tensors c_{ijkl}, d_{kij} and ε_{ij} playing role of the material constants. ¿From the conditions of the thermodynamic stability, tensors c_{ijkl} and ε_{ij} have to be symmetric and positive definite (see, e.g., [4]).

Let us have the piezolectric resonator characterized by proper material tensors. The density of the material is ϱ. We denote the volume of the resonator as Ω and its boundary as Γ. There are two differential equations governing the behavior of a piezoelectric continuum - the Newton's law of motion (3) and the quasistatic approximation to the Maxwell's equation (4) (see [3])

$$\varrho \frac{\partial^2 \tilde{u}_i}{\partial t^2} = \frac{\partial \mathrm{T_{ij}}}{\partial x_j} \qquad i = 1,2,3, \qquad x \in \Omega, \quad \mathrm{t} \in (0, \mathrm{T}), \tag{3}$$

$$-\nabla \cdot \mathbf{D} = -\frac{\partial \mathrm{D_j}}{\partial x_j} = 0. \tag{4}$$

We will call them *elastic* and *electric* equations. If we replace **T** and **D** in (3) and (4) with the expressions (1) and (2), we obtain

$$\varrho \frac{\partial^2 \tilde{u}_i}{\partial t^2} = \frac{\partial}{\partial x_j}\left(c_{ijkl} \cdot \frac{1}{2}\left[\frac{\partial \tilde{u}_k}{\partial x_l} + \frac{\partial \tilde{u}_l}{\partial x_k}\right] - d_{ijk} \cdot \frac{\partial \tilde{\varphi}}{\partial x_k}\right) \quad i = 1,2,3, \tag{5}$$

$$0 = \frac{\partial}{\partial x_k}\left(-d_{kij} \cdot \frac{1}{2}\left[\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right] + \varepsilon_{kj} \cdot \frac{\partial \tilde{\varphi}}{\partial x_j}\right). \tag{6}$$

Initial conditions, Dirichlet boundary conditions (on the part of the boundary $\Gamma_1 \subset \Gamma$) and Neumann boundary conditions (on $\Gamma_2 \subset \Gamma$) are added:

$$\begin{aligned}
\tilde{u}_i(.,0) &= u_i, \\
\tilde{u}_i &= 0 \qquad i = 1,2,3 \qquad \text{on} \quad \Gamma_1 \times (0, \mathrm{T}), \\
\mathrm{T_{ij}} n_j &= 0 \qquad i = 1,2,3 \qquad \text{on} \quad \Gamma_2 \times (0, \mathrm{T}), \\
\tilde{\varphi}(.,0) &= \varphi, \\
\tilde{\varphi} &= \varphi_D \qquad \text{on} \quad \Gamma_1 \times (0, \mathrm{T}), \\
\mathrm{D_k} n_k &= 0 \qquad \text{on} \quad \Gamma_2 \times (0, \mathrm{T}).
\end{aligned} \tag{7}$$

We don't consider loading with other outer, e.g. mechanical, forces. Assuming the harmonic electric potential incoming to the resonator, we can separate time and space variables,

$$\tilde{\varphi} = \varphi(x,y,z)\cos \omega t, \tag{8}$$

where ω is the frequency of voltage and φ_0 is its amplitude. In (6), displacement $\tilde{\mathbf{u}}$ is determined by electric potential $\tilde{\varphi}$, which generates the oscillations. The changes of displacement delay to generating, but if we assume the steady undamped oscillation, we can expect, that the resonator will harmonic oscillate with the same initiating frequency ω, and separate time and space variables,

$$\tilde{\mathbf{u}} = \mathbf{u}(x, y, z) \cos \omega t, \tag{9}$$

where $\mathbf{u}$ is the amplitude of oscillations. With respect to

$$\varrho \frac{\partial^2 \tilde{u}_i}{\partial t^2} = -\omega^2 \varrho \tilde{u}_i,$$

substituting (9) and (8) into (5) and (6) gives the modified versions of elastic and electric equations, now for the unknown amplitudes $\mathbf{u}$ and φ,

$$-\omega^2 \varrho \cdot u_i = \frac{\partial}{\partial x_j}\left(c_{ijkl} \cdot \frac{1}{2}\left[\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}\right] - d_{ijk} \cdot \frac{\partial \varphi}{\partial x_k}\right) \quad i = 1, 2, 3, \tag{10}$$

$$0 = \frac{\partial}{\partial x_k}\left(-d_{kij} \cdot \frac{1}{2}\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] + \varepsilon_{kj} \cdot \frac{\partial \varphi}{\partial x_j}\right), \tag{11}$$

with the boundary conditions

$$\begin{aligned} u_i &= 0 \qquad i = 1, 2, 3 \qquad \text{on} \quad \Gamma_1, \\ \varphi &= \varphi_D \qquad \text{on} \quad \Gamma_1, \\ \mathrm{T_{ij}} n_j &= 0 \qquad i = 1, 2, 3 \qquad \text{on} \quad \Gamma_2, \\ \mathrm{D_k} n_k &= 0 \qquad \text{on} \quad \Gamma_2. \end{aligned} \tag{12}$$

1.2. Point of interest

In (10)-(12), the problem of steady undamped harmonic oscillations is defined. Now, our goal is to find the eigenfrequencies of the system (10)-(11) - such real number ω, for which exist functions $\mathbf{u}$ and φ, satisfying (10)-(11). Under the term *resonance frequency* we will mean these eigenfrequencies. If we define the operators $\mathbf{A}$ and $\mathbf{B}$ as

$$\mathbf{A}(\mathbf{u}, \varphi) = -\frac{1}{\varrho}\frac{\partial}{\partial x_j}\left(c_{ijkl} \cdot \frac{1}{2}\left[\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}\right] - d_{ijk} \cdot \frac{\partial \varphi}{\partial x_k}\right)$$

$$\mathbf{B}(\mathbf{u}, \varphi) = \frac{\partial}{\partial x_k}\left(d_{kij} \cdot \frac{1}{2}\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] - \varepsilon_{kj} \cdot \frac{\partial \varphi}{\partial x_j}\right),$$

then the resonance frequencies are the roots of the eigenvalues λ, $\lambda = \omega^2$, from

$$\mathbf{A}(\mathbf{u}, \varphi) = \lambda \mathbf{u}, \tag{13}$$

where the electric potential φ is linked with the displacement $\mathbf{u}$ by the condition

$$\mathbf{B}(\mathbf{u}, \varphi) = 0. \tag{14}$$

The eigenvector $\mathbf{u}$ corresponding to the eigenvalue λ describes the mode of the oscillations. To identify the mode of oscillations, it is additional task needed for sufficient description of the behavior of the piezoelectric resonator. We discretize the problem, using finite element method (FEM). Discretization of the problem then leads to a large sparse linear algebraic system, which defines the generalized eigenvalue problem. Resonance frequencies are subsequently found by solving this algebraic problem.

2. NUMERICAL SOLUTION

2.1. Weak formulation

Before we discretize the problem (10)-(12) and use the finite element method to get the linear system, we establish the *weak formulation* of the problem (10)-(12). Equations (10), resp. (11) represent energy, resp. electric flux conservation. These equations are elliptic partial differential equations and its exact solution must have continuous derivations to the 2nd order in Ω. In fact, the requirements to the functions smoothness is too strong. Usually, the energy conservation laws are described with integral principles known from theoretical physic. So the idea is to transform equations (10), (11) "back" to the integral form, containing derivations of lower orders than in original equations. This process, described in next paragraph, is called weak formulation. We require the *weak solution* to fulfil integral equations in certain functional sence.

We deal with the standard weak formulation, derived e.g. in [5]. We consider bounded domain Ω with Lipschitzian border Γ. Let $L_2(\Omega)$ be the Lebesgeue space of functions square integrable in Ω, $\int_\Omega |f|^2 \ll +\infty$, with scalar product $(f,g)_\Omega = \int_\Omega fg d\Omega$. Further, let $C^{(\infty)}(\overline{\Omega})$ be functions, which, including derivatives of all orders, are continuous in $\overline{\Omega}$. $C_0^{(\infty)}(\overline{\Omega}) \subset C^{(\infty)}(\overline{\Omega})$ contains the functions with compact support.

Further, let $W_2^{(1)}(\Omega)$ be so-called *Sobolev space*, made of functions from $L_2(\Omega)$, which have so-called *generalized derivatives* square integrable in Ω - there exist some function $u^i \in L_2(\Omega)$ and the identity (15) is fulfilled,

$$\int_\Omega u^i \psi d\Omega = -\int_\Omega u \frac{\partial \psi}{\partial x_i} d\Omega \quad \forall \psi \in C_0^{(\infty)}(\overline{\Omega}). \tag{15}$$

For functions lying in $C^{(\infty)}(\overline{\Omega})$, generalized derivatives are its classical derivatives. To express values of function $u \in W_2^{(1)}(\Omega)$ on the border Γ, the *trace* of function u is established. If $u \in C^{(\infty)}(\overline{\Omega})$, its values on the border, we denote them $u(S)$, are uniqely determined. For function $u(S)$ is the trace of the function $u \in C^{(\infty)}(\overline{\Omega})$. For function $u \in W_2^{(1)}(\Omega) \backslash C^{(\infty)}(\overline{\Omega})$, there exist a function sequence $(u_n) \subset W_2^{(1)}(\Omega)$ such that $u = \lim_{n\to\infty} u_n$, and its trace can be defined as the function $u(S) \in L_2(\Omega)$, which is a limit of the suqeuence of traces

$$u(S) = \lim_{n\to\infty} u_n(S) \quad \text{in} \quad L_2(\Omega).$$

The traces can be defined, in the same way, also for the derivatives of the functions from $W_2^{(1)}(\Omega)$.

Now, we define,

$$V(\Omega) = \{v | v \in W_2^{(1)}(\Omega), \quad v|_{\Gamma_1} = 0 \text{ in the sence of traces}\},$$

the subspace of $W_2^{(1)}(\Omega)$, made of functions, which traces fulfil the homogenous boundary conditions.

We derive the weak formulation in the standard way (see e.g. [5]). We multiply the equations (5) with testing functions $w_i \in V(\Omega)$, summarize and integrate them over Ω. As well, we multiply the equation (6) with testing function $\phi \in V$ and integrate it over Ω. Using Green formula, we obtain the integral equalities (integrals over the borders are denoted with sharp brackets)

$$\begin{aligned} \left(c_{ijkl} \cdot \frac{1}{2}\left[\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}\right], \frac{\partial w_i}{\partial x_j} \right)_\Omega &- \left(\varrho\omega^2 u_i, w_i \right)_\Omega \\ - \left(d_{ijk} \cdot \frac{\partial \varphi}{\partial x_j}, \frac{\partial w_i}{\partial x_j} \right)_\Omega &= \left\langle \mathrm{T_{ij}} \cdot n_j, w_i \right\rangle_{\Gamma_2}, \end{aligned} \tag{16}$$

$$-\left(d_{jik} \cdot \frac{1}{2}\left[\frac{\partial u_i}{\partial x_k} + \frac{\partial u_k}{\partial x_i}\right], \frac{\partial \phi}{\partial x_j} \right)_\Omega + \left(\varepsilon_{ji} \cdot \frac{\partial \varphi}{\partial x_i}, \frac{\partial \phi}{\partial x_j} \right)_\Omega = \left\langle D_N, \phi \right\rangle_{\Gamma_2}. \tag{17}$$

Due to the symmetry of material tensors, we can modify equations (16) and (17),

$$\begin{aligned} \left(c_{ijkl} \cdot \frac{1}{2}\left[\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k}\right], \frac{1}{2}\left[\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right] \right)_\Omega &- \left(\varrho\omega^2 u_i, w_i \right)_\Omega \\ - \left(d_{ijk} \cdot \frac{\partial \varphi}{\partial x_j}, \frac{1}{2}\left[\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right] \right)_\Omega &= \left\langle \mathrm{T_{ij}} \cdot n_j, w_i \right\rangle_{\Gamma_2}, \end{aligned} \tag{18}$$

$$-\left(d_{jik} \cdot \frac{1}{2}\left[\frac{\partial u_i}{\partial x_k} + \frac{\partial u_k}{\partial x_i}\right], \frac{\partial \phi}{\partial x_j} \right)_\Omega + \left(\varepsilon_{ji} \cdot \frac{\partial \varphi}{\partial x_i}, \frac{\partial \phi}{\partial x_j} \right)_\Omega = \left\langle D_N, \phi \right\rangle_{\Gamma_2}. \tag{19}$$

Let us denote

$$\mathrm{S}_{ij} = \frac{1}{2}\left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right], \quad \mathrm{R}_{ij} = \frac{1}{2}\left[\frac{\partial w_i}{\partial x_j} + \frac{\partial w_j}{\partial x_i}\right], \qquad i,j = 1,2,3.$$

Substitution of boundary conditions (12) into equations (18) and (19) gives (integrals over the border are zeros)

$$\left(c_{ijkl} \cdot \mathrm{S}_{kl}, \mathrm{R}_{ij}\right)_\Omega - \left(\varrho\omega^2 u_i, w_i\right)_\Omega - \left(d_{ijk} \cdot \frac{\partial \varphi}{\partial x_j}, \mathrm{R}_{ij}\right)_\Omega = 0, \tag{20}$$

$$-\left(d_{jik} \cdot \mathrm{S}_{ik}, \frac{\partial \phi}{\partial x_j}\right)_\Omega + \left(\varepsilon_{ji} \cdot \frac{\partial \varphi}{\partial x_i}, \frac{\partial \phi}{\partial x_j}\right)_\Omega = 0. \tag{21}$$

Weak solution: Let $\mathbf{u}_D = (u_1, u_2, u_3) \in [W_2^1(\Omega)]^3, \quad \varphi_D \in W_2^1(\Omega)$ satisfy the Dirichlet boundary conditions (in the weak sence). Further, let $\mathbf{u}_0 = (u_1, u_2, u_3) \in [W_2^1(\Omega)]^3, \quad \varphi_0 \in W_2^1(\Omega)$ be functions, for which equalities (20) and (21) are observed for all choices of testing functions $\mathbf{w} = (w_1, w_2, w_3) \in [V(\Omega)]^3, \quad \phi \in V(\Omega)$. Then we define the weak solution of the problem (10)-(12) as

$$\mathbf{u} = \mathbf{u}_D + \mathbf{u}_0, \quad \varphi = \varphi_D + \varphi_0.$$

2.2. Discretization

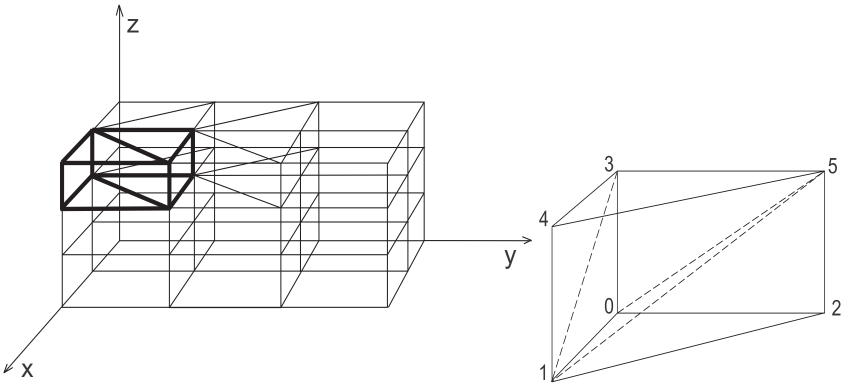


Figure 1: Discretization of the crystal into layers and prismatic elements and an example of division of an prismatic elements into three simplex elements 0125, 0153 a 1534

Finite element method looks for a certain approximation of weak solution. We disretize the area Ω into the set of *finite elements*, where special base functions are established. Now, weak solution as the linear combination of these base functions is looked for. The part $\mathbf{u}_D, \varphi_D$ of the weak solution, satisfying the Dirichlet boundary conditions, can be explicitly expressed in the linear system, resulting from discretization of the problem (20), (21). It will be introduced later. For computing an aproximation of the homogenous part of the weak solution of our problem, we divide the area Ω (which is the volume of the resonator), in two steps, to the finite set E^h of disjoint tetrahedrons covering the volume (first part - shown in the fig. 1 left - is the division into the layers and prizmatic elements, second part - fig. 1 right- is the division of the prizmatic elements into the tethrahedrons),

$$\Omega \sim \Omega^h = \bigcup_{e \in E^h} e, \quad \bigcup_{j \in J} \overline{e_j} = \overline{\Omega}.$$

For each element $e \in E^h$, we define the function space $V^h(e)$ and its basis $\Phi^h(e)$

$$\begin{aligned} V^h(e) &= \{\phi^h | supp(\phi^h) \subset e, \quad \phi^h \in W_2^1(e), \quad \phi^h|_{\partial e} = 0\}, \\ \Phi^h(e) &= \{\phi_i^e(x,y,z) | i = 1,2,3,4\}. \end{aligned}$$

Base functions are defined by its values at the nodes s^j of the element and have to satisfy

$$\phi_i^e(s^j) = \delta_{ij}, \qquad i, j = 1, 2, 3, 4.$$

On each simplex, four linear base functions are established. The approximations of the electric potential and displacement in whole Ω^h are

$$\begin{aligned} u_i^h(x) &= \sum_{\phi_j^h \in \Phi^h} u_i^j \phi_j^h(\mathbf{x}), \quad u_i^j \in \mathbf{R}, \quad \mathbf{x} \in \Omega, \quad i = 1, 2, 3, \\ \varphi^h(\mathbf{x}) &= \sum_{\phi_j^h \in \Phi^h} \varphi^j \phi_j^h(\mathbf{x}), \quad \varphi^j \in \mathbf{R}, \quad \mathbf{x} \in \Omega, \end{aligned} \tag{22}$$

where Φ^h denotes the set of all base functions. These approximations are piecewise linear on each element. Coefficients in the linear combination are the values of the functions $\mathbf{u}$ and φ in the nodes of division. Let the nodes of the division and proper base functions be numbered $(\phi_1^h, ..., \phi_r^h)$. We substitute the approximations (22) into integral equalities (20) and (21). We require to them to be fulfilled for all base functions ϕ_s^h, $s \in \hat{r}$,

$$\left(c_{ijkl} \cdot \mathrm{S}^{\mathrm{h}}_{\mathrm{kl}}, \mathrm{R}^{\mathrm{h}}_{\mathrm{ij}}\right)_\Omega - \left(\varrho\omega^2 u_i^h, \phi_s^h\right)_\Omega - \left(d_{ijk} \cdot \frac{\partial \varphi^h}{\partial x_j}, \mathrm{R}^{\mathrm{h}}_{\mathrm{ij}}\right)_\Omega = 0,$$

$$-\left(d_{jik} \cdot \mathrm{S}^h_{ik}, \frac{\partial \phi_s^h}{\partial x_j}\right)_\Omega + \left(\varepsilon_{ji} \cdot \frac{\partial \varphi^h}{\partial x_i}, \frac{\partial \phi_s^h}{\partial x_j}\right)_\Omega = 0.$$

To satisfy the above equations, the system of linear algebraic equations has to be fulfiled. The system has a block shape

$$\left(\begin{array}{ccc} \mathbb{K}_{11} & .. & \mathbb{K}_{1r} \\ \mathbb{K}_{21} & .. & \mathbb{K}_{2r} \\ ... & .. & ... \\ ... & .. & ... \\ \mathbb{K}_{r1} & .. & \mathbb{K}_{rr} \end{array} -\omega^2 \begin{array}{ccc} \mathbb{M}_{11} & .. & \mathbb{M}_{1r} \\ \mathbb{M}_{21} & .. & \mathbb{M}_{2r} \\ ... & .. & ... \\ ... & .. & ... \\ \mathbb{M}_{r1} & .. & \mathbb{M}_{rr} \end{array} \left| \begin{array}{ccc} \mathbb{P}^{\mathrm{T}}_{11} & ... & \mathbb{P}^{\mathrm{T}}_{r1} \\ \mathbb{P}^{\mathrm{T}}_{12} & ... & \mathbb{P}^{\mathrm{T}}_{r2} \\ ... & ... & ... \\ ... & ... & ... \\ \mathbb{P}^{\mathrm{T}}_{1r} & ... & \mathbb{P}^{\mathrm{T}}_{rr} \end{array} \right. \right. \atop \left. \begin{array}{ccc} \mathbb{P}_{11} & ... & \mathbb{P}_{1r} \\ \mathbb{P}_{21} & ... & \mathbb{P}_{2r} \\ \vdots & \dots & \vdots \\ \mathbb{P}_{r1} & ... & \mathbb{P}_{rr} \end{array} \qquad\qquad \left| \begin{array}{ccc} \mathbb{E}_{11} & ... & \mathbb{E}_{1r} \\ \mathbb{E}_{21} & ... & \mathbb{E}_{2r} \\ \vdots & \dots & \vdots \\ \mathbb{E}_{r1} & ... & \mathbb{E}_{rr} \end{array} \right. \right) \begin{pmatrix} u_1^1 \\ u_2^1 \\ u_3^1 \\ ... \\ u_1^r \\ u_2^r \\ u_3^r \\ \varphi^1 \\ \varphi^2 \\ ... \\ \varphi^r \end{pmatrix} = \begin{pmatrix} 0 \\ ... \\ ... \\ ... \\ ... \\ ... \\ ... \\ ... \\ ... \\ 0 \end{pmatrix}, \tag{23}$$

where $\mathbb{K}_{pq}, \mathbb{M}_{pq} \in \mathbf{R}^{3,3}$, $\mathbb{P}_{pq} \in \mathbf{R}^{1,3}$, $\mathbb{E}_{pq} \in \mathbf{R}$. The process of derivation of the system matrix is in detail described in [6]. Let us write the system (23) as

$$\begin{pmatrix} \mathbb{K} - \omega^2 \mathbb{M} & -\mathbb{P}^{\mathrm{T}} \\ -\mathbb{P} & \mathbb{E} \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{24}$$

The submatrix $\mathbb{K} \in \mathbf{R}^{3r,3r}$ is the elastic matrix, $\mathbb{M} \in \mathbf{R}^{3r,3r}$ is the massmatrix, $\mathbb{P} \in \mathbf{R}^{r,3r}$ is the piezoelectric matrix and $\mathbb{E} \in \mathbf{R}^{r,r}$ is the electric matrix. Due to the symmetry and positive definiteness of material tensors, the matrices $\mathbb{K}, \mathbb{M}, \mathbb{E}$ are symmetric and positive definite. Further, the whole system matrix (24) is symmetric. $\mathrm{U} \in \mathbf{R}^{3r}$, resp. $\mathrm{V} \in \mathbf{R}^r$ are values (of amplitudes) of displacement, resp. electric potential at the nodes of division. The matrices are sparse - the pq-th block of each submatrix is nonzero only if p-th and q-th node of division have common edge.

2.3. Algebraic problem

After dicretization of the problem (10)-(12), we have obtained the system (24), where ω is the independent parameter. As was mentioned in the paragraph [1.2], the resonance frequencies are the eigenvalues from the problem (13)-(14). This eigenproblem corresponds to the generalized eigenvalue problem,

$$\begin{pmatrix} \mathbb{K} & -\mathbb{P}^{\mathrm{T}} \\ -\mathbb{P} & \mathbb{E} \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} = \lambda \begin{pmatrix} \mathbb{M} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} \tag{25}$$

and $\omega = \sqrt{\lambda}$ is the resonance frequency, if there exist real number λ and real nonzero vectors U, V and (25) is fulfilled. The part U of the eigenvector describes the shape of the oscillations and it is called the *mode* of the oscillation. The set of eigenfrequencies and proper eigenvectors describes the natural oscillation of the piezoelectric system.

Equation (25) can be written in two equations

$$\begin{aligned} \mathbb{K}\mathrm{U} - \mathbb{P}^{\mathrm{T}}\mathrm{V} &= \lambda\mathbb{M}\mathrm{U}, & (26)\\ \mathbb{P}\mathrm{U} - \mathbb{E}\mathrm{V} &= 0. & (27) \end{aligned}$$

The matrix $\mathbb{E}$ is positive definite and therefore it is invertible. From (27) we have

$$\mathrm{V} = \mathbb{E}^{-1}\mathbb{P}\mathrm{U}$$

and (26) can be written as

$$[\mathbb{K} - \mathbb{P}^{\mathrm{T}}\mathbb{E}^{-1}\mathbb{P}]\mathrm{U} = \lambda\mathbb{M}\mathrm{U}. \tag{28}$$

So the generalized eigenproblem (25) is equivalent to the generalized symmetric definite eigenvalue problem (28). In practice, we are not interested in all eigenvalues. The basic modes of oscillation belong to the first few of them.

2.4. Boundary conditions

We deal with Dirichlet boundary conditions (12) for displacement and electric potential. The introduction of the boundary conditions is sketched on the fig. 2. First is the case of homogenous boundary conditions for displacement $\mathbf{u}$. Let there be in some nodes prescribed zero displacements (on the fig.2 marked with gray color). Then proper columns of the matrix (marked with gray color) are multiplied by zeros and can be eliminated. So can be eliminated the prescribed variables from the vector of unknowns. Now, the number of equation is bigger than the number of unknows, thus the rows (marked with gray color) belonging to the known variables can be eliminated. The qualities of the matrix remain the same, only its size decreases.

In the case of nonhomogenous Dirichlet boundary conditions for electric potential, there are some differences. The part of the vector with prescribed values is marked with the grid. The proper columns of the matrix are multiplied by prescribed vlaues and the resulting vector forms the right side of the linear system. The rows (marked with the grid) belonging to the known variables can be eliminated.

The linear system with right side results, with deflated matrix,

$$\begin{pmatrix} \mathbb{K} - \omega^2\mathbb{M} & -\mathbb{P}^{\mathrm{T}} \\ -\mathbb{P} & \mathbb{E} \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} = \begin{pmatrix} \mathrm{R}_1 \\ \mathrm{R}_2 \end{pmatrix}. \tag{29}$$

The system matrix has the same properties as the matrix of the original system (24). In (29), at first the generalized eigenvalue problem, analogic to (25), has to be solved. Then, for given eigenvalue $\hat{\lambda}$, the particular solution $\mathrm{W} = (\mathrm{W}_1, \mathrm{W}_2)^{\mathrm{T}}$ of the system

$$\begin{pmatrix} \mathbb{K} - \hat{\lambda}\mathbb{M} & -\mathbb{P}^{\mathrm{T}} \\ -\mathbb{P} & \mathbb{E} \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} = \begin{pmatrix} \mathrm{R}_1 \\ \mathrm{R}_2 \end{pmatrix}$$

has to be found, being the linear combination of all eigenvectors,

$$\mathrm{W} = \mathbb{P}\mathrm{Z},$$

where $\mathbb{P}$ is the matrix, which columns are eigenvectors, Z is the real vector of the coefficients of the linear combination.

2.5. Computer implementation

For discretization and compilation of the global matrix, we have developed our own code. For solving the eigenvalue problem (26), we use the procedures from the Lapack++, resp. Arpack++ library, available on

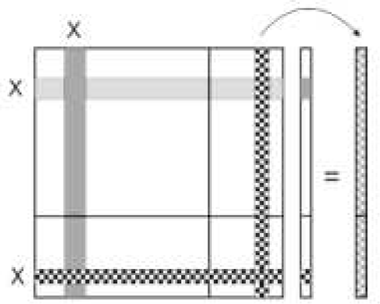


Figure 2: Introduction of boundary conditions into the linear system

the internet. ¿From Lapack++ (see [3]), we use algorithm based on generalized Schur decomposition. This algorithm solves the complete eigenvalue problem. ¿From Arpack++ (see [4]), we use algorithm based on shift-invert method combined with LU factorization. This algorithm, in contrast to Lapack++ code, solves the partial eigenvalue problem and deal with the fact, that matrices are sparse.

2.6. New approach planned for solving the algebraic problem

The matrix from (24) has similar scheme as so called *system matrix* resulting from problems of computing *zeros* of a linear multivariable system (see [7] or [8]),

$$\begin{pmatrix} \lambda\mathbb{I} - \mathbb{A} & -\mathbb{B} \\ \mathbb{C} & \mathbb{D} \end{pmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{V} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

where V is the input vector and X is the state vector of the system. The *zeros* are the generalized eigenvalues λ of the matrix pencil (30),

$$\begin{pmatrix} \lambda\mathbb{I} - \mathbb{A} & -\mathbb{C} \\ \mathbb{B} & \mathbb{D} \end{pmatrix}. \tag{30}$$

Equations (26)-(27) can be premultiplied by $\mathbb{M}^{-1}$ and eigenvalue problem (25) is equivalent to the problem

$$\begin{pmatrix} \lambda\mathbb{I} - \mathbb{M}^{-1}\mathbb{K} & \mathbb{M}^{-1}\mathbb{P}^{\mathrm{T}} \\ \mathbb{M}^{-1}\mathbb{P} & \mathbb{M}^{-1}\mathbb{E} \end{pmatrix} \begin{pmatrix} \mathrm{U} \\ \mathrm{V} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix},$$

with the matrix having the similar scheme as system matrix (30) (it is a special symmetric case of (30)). Also the characteristic of the linear multivariable systems corresponds, in certain sence, to the physical base of our problem (electric potential plays the role of input vector, displacement is state vector of the system).

For computing generalized eigenvalues of (30) an algorithm based on the QZ method is developed and in detail discussed in [7]. We propose to use this algorithm for solving the eigenvalue problem (25).

3. TESTING PROBLEM

The designed FEM model was calibrated and verified on the longitudinally vibrating narrow quartz XYt-j -cut rods (for j = $0^{\mathrm{o}} - 5^{\mathrm{o}}$) with equivalent thickness. Both large sides of resonator are covered by silver electrodes. The resonator is fixed in the center of its length, thus the problem is symmetric and we can solve it for one half of the resonator (this problem was publicated in [1]).

Boundary conditions: $\varphi = \varphi_{def}$ at the electrodes, $\mathbf{u} = 0$ at the nodal line of odd vibrations (in the center large sides). Zero displacements are also entered on the planes going through the nodal line of odd vibrations. These planes are supposed in two modifications: $\mathbf{u} = 0$ in the plane normal to the length of resonator, or $\mathbf{u} = 0$ in the nodal plane of longitudinal vibrations. The definition of the second plane depends on the cut (see [2]) and the plane is not exactly normal to the length of the resonator. This second condition represents more accurate the physical reality.

3.1. Numerical realisation

The resonator was divided into prismatic elements (fig. 3) and then each of the prismatic element was divided into four tetrahedrons. For eigenvalue problem, Lapack++, resp. Arpack++ solvers were used. The computation was repeated several times with refined meshes. Computed frequencies of longitudinally

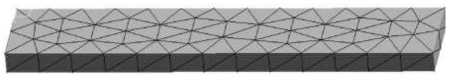


Figure 3: The mesh - prismatic elements

vibrations are compared with the measured frequencies (publicated in [1]) in the table below.

measured average value (Hz)	deviation max.(Hz)	deviation min.(Hz)	α	computed values
67846	$+123$	-114	$0°$	$68,55 \cdot 10^3$
68653	$+67$	-51	$2°$	$68,82 \cdot 10^3$
70205	$+60$	-119	$5°$	$69,05 \cdot 10^3$

In the figure (4), the convergence of computed frequencies (to the value 68.82 kHz), in dependance on number of elements, is shown. The Arpack++ code was rather faster then the Lapack++ code.

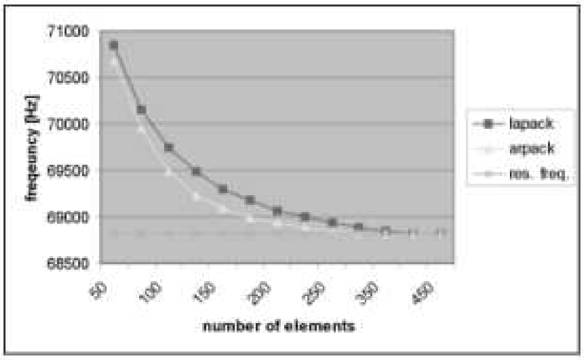


Figure 4: Convergence of the resonance frequency to the value $68,82 \cdot 10^3$ Hz

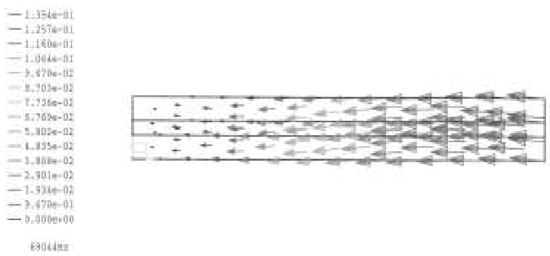


Figure 5: Demonstration of longitudinally vibrations

4. CONCLUSION

The mathematical model for computing the resonance frequencies of the piezoelectric resonator has been built. The results of the described model approximate well the measured results for tested, simply shaped (rod or slide), resonators. It seems that our model can have real application, e.g. in desining shapes of the resonators vibrating with required frequencies. The correspondence with the measured results increases for more fine meshes. Of course, it involves to solve the eigenvalue problems with very large matrices. Nowadays, the use of the more sofisticated numerical algorithm (mentioned in the paragraph [2.6]) for solving the eigenvalue problem is proposed.

5. Acknowledgement

This project was supported by Ministry of Education of the Czech Republic, project code MSM 242200002.

References

[1] Tichý J., Zelenka J.: "Podélně a plošně střižně kmitající piezoelektrické rezonátory ze synthetického křemene", *Čs. Čas. Fys.*, vol. 10, pp. 328 - 332, 1960.

[2] Zelenka J.: "Piezoelektrické rezonátory a jejich použití v praxi", Academia, Praha, 1981.

[3] Milsom R. F., Elliot D. T., Terry Wood S., Redwood M.: "Analysis and Design of Couple Mode Miniature Bar Resonator and Monolothic Filters", *IEEE Trans Son. Ultrason.*, vol. 30, pp. 140- 155, 1983.

[4] Semenčenko V.K.: "Izbrannye glavy teoretičeskoj fiziki", Voennogo izdatelstvo Ministerstva oborony SSSR, Leningrad, 1960.

[5] Rektorys K.: "Variační metody", Academia Praha, 1989.

[6] P. Rálek: "Modelování piezoelektrických jevů", Diploma thesis, FJFI ČVUT, Praha, 2001.

[7] Emami-Naeini A., Van Dooren P.: "Computation of Zeros of Linear Multivariable Systems", *Automatica*, vol. 18 no. 4, pp. 415-430, 1982.

[8] Macfarlane A. G. J., Karcanias N.: "Poles and zeros of linear multivariable systems", *Int. J. Control*, vol. 24 no. I, pp. 33-74, 1976.

Validation of the coronary heart disease prediction

doktorand:
JINDRA REISSIGOVÁ
Institute of Computer Science AS CR,
EuroMISE Centre – Cardio, Prague
reissigova@euromise.cz

školitel:
JANA ZVAROVÁ
Institute of Computer Science AS CR,
EuroMISE Centre – Cardio, Prague
zvarova@euromise.cz

obor studia:
Biomedical Informatics

Abstract

The aim was to validate the estimate of coronary heart disease risk derived from the Framingham Heart Study (FHS). In the 20-year intervention study of the risk factors of atherosclerosis (STULONG), the accuracy of the Framingham coronary heart disease risk was evaluated by the Receiver Operating Characteristic (ROC) curve and chi-square goodness of the fit test. During 10 years from the entry into STULONG, 141 CHD were predicted and 116 observed among 1 125 men aged of 38–53 years at the entry into STULONG in 1975-1979. No significant difference was found between the expected and observed numbers. According to the ROC curve, the Framingham coronary heart disease risk classified the men from STULONG into those with and without developing CHD within 10-year period with 74% accuracy. The work suggested that the Framingham risk seemed to be a fair indicator of a coronary heart disease for men of STULONG.

1. Introduction

The cardiovascular diseases (CVD) is the name for the group of disorders of the heart and blood vessels and include for instance coronary heart disease (CHD) and cerebrovascular disease (stroke). CVD are ones of the most common diseases in the Czech Republic and most of the developed world. In 1999, CVD contributed to a third of global deaths, low and middle income countries contributed to 78 % of CVD deaths. By 2010 CVD is estimated to be the leading cause of death in developing countries. The risk factors of CVD include eg. advancing age, cigarette smoking, hypertension, hypercholesterolaemia, obesity, physical inactivity, unhealthy diet, see `http://www.who.int/cardiovascular_diseases/priorities/en/`.

The aim is to develop the global preventive strategy to reduce the incidence and mortality of CVD among high risk population by effectively reducing CVD risk factors. The task is how to identify person free of CVD with high probability (risk) of developing CVD within a certain time period. Epidemiologists, statisticians and other health workers have been working on methods of producing a probability (an absolute risk) estimate of developing CVD [1], [2], [3], [7], [9], [10], [11].

In the Czech Republic, one of the most used predictive models estimating the probability of developing CHD is the model of the Framingham Heart Study investigators [1]. Subsequent Framingham models have been based on larger and more recent follow-up and used better predictive variables and more sophisticated statistical methods [3].

The objective of this paper was to validate the Framingham coronary heart disease prediction [1] in the longitudinal study of the risk factors of atherosclerosis (RFA) launched in Prague in the year 1975.

2. Materials and methods

The longitudinal study of the risk factors of atherosclerosis (STULONG)

STULONG is the intervention prime preventive study with multiple risk factor intervention, which was conducted by 2nd Dep. of Internal Medicine, 1st Faculty of Medicine and General Faculty Hospital, Charles University in Prague 2 in 1975–1999. Originally, STULONG was a part of a national wide study "National primary preventive multifactorial study of myocardial infarction and stroke" in former Czechoslovakia (in the study more than 10 000 subjects should be included) [6].

In 1975 total 2370 men aged 38–49 living in the 2nd district in the centre of Prague (Prague 2) were randomly selected from list of electors. It was the 50 % sample of men of that age who were living in Prague 2 in 1975. Of 2370 invited men, 1417 (59.8 %) men answered the invitation and underwent entry examination in 1975–1979. Entry questionnaire included questions on demographic and personal data (marital status, education, working physical activity, leisure physical activity, smoking, alcohol drinking, coffee drinking, tee drinking, personal and family anamnesis, chest pain, lower limbs pain, breathlesness) and results of physical (height, weight, diastolic blood pressure (BP), systolic BP, skinfolds), laboratory (cholesterol level, triglyceridy, uric acid) and ECG (electrocardiography) measurements.

According to health status and occurrence of RFA (see Table 1) at the entry into the study, each man was classified into one of three groups (normal, risk and pathological) differing in way of multiple risk factor intervention in the 20–year follow-up, see Figure

Table 1: The risk factors of atherosclerosis at the entry into the study in1975–1979

Positive family history	death on the atherosclerotic diseases before the age of 65 years in the parents
Obesity	Brocca index (BI)$\geq$115 %, where BI=weight[kg]$/$(height[m] $-$ 100) $\cdot$ 100 %
Smoking	$\geq$15 cigarettes daily; or non-smoker less than one year and $\geq$15 cigarettes daily before
Hypertension	blood pressure $\geq$160 and/or 95 mmHg in two of three measurements; or hypertension in anamnesis
Hypercholesterolaemia	total cholesterol $\geq$260mg % (6.7 mmol/l)

Normal Group (NG) included men without any RFA mentioned in Table 3, without CVD, without diabetes mellitus, without other serious disease not enabling long term follow-up and without pathological finding on ECG curve at the entry into the study. NG was randomly divided into two groups: normal group regularly examined (NGE, $n = 40$) and normal group regularly unexamined (NGN, $n = 236$). NGE was yearly examined by specialists from 2nd Dep. of Internal Medicine. If RFA (obesity, smoking, hypertension, hyperlipaemia) was detected specialists initiated pharmacological and non pharmacological (feeding habits, physical activity, smoking etc.) intervention of RFA. If detected RFA, man was examined in 2nd Dep. of Internal Medicine as needed. The control questionnaire was filled out once a year. The control questionnaire was consistent with entry questionnaire except question on family anamnesis excluded in the control questionnaire, and question on feeding habits extra included in the control questionnaire. NGE was examined by specialists from 2nd Dep. of Internal Medicine once in 8th-13th year from the entry into the study. If CVD was detected, man was offset in the pathologic group, i.e. in the next years man was not investigated within the study.

Risk group (RG) included men with at least one of RFA (Table 3), without CVD, diabetes mellitus and other

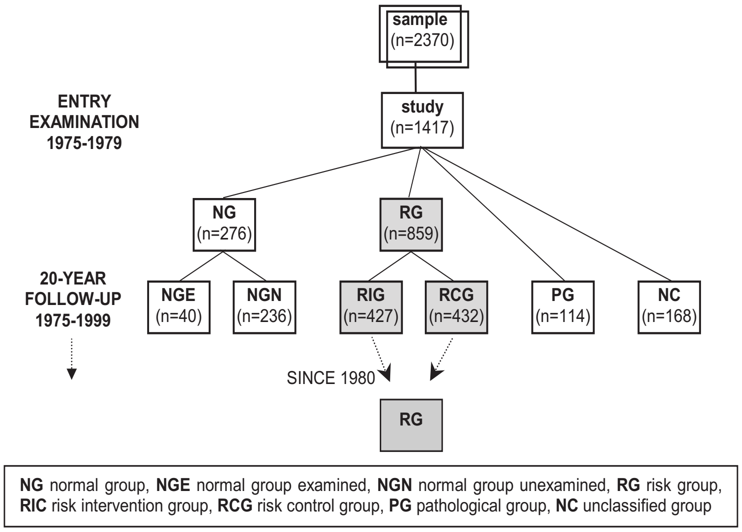


Figure 1: Design of the intervene prime preventive study of atherosclerosis (STULONG)

serious disease not enabling long term follow-up and without pathological finding on ECG curve at the entry into the study. RG was randomised into two subgroups: risk intervention group (RIG, $n = 427$) and risk control group (RCG, $n = 432$). Pharmacological and non-pharmacological intervention of RFA in RIG was performed by specialists from 2nd Dep. of Internal Medicine, RCG was under health care of general practitioners. In both groups, the control questionnaire introduced above (see Normal group) was filled out once a year by the specialist from 2nd Dep. of Internal Medicine. Since early eighties of the last century, the groups RIG and RCG had been melting, mainly for ethic reasons [8].

Pathological group (PG) included men with CVD, diabetes mellitus or other serious disease not enabling long term follow-up or with pathological finding on ECG curve at the entry into the study.

The Framingham Heart study (FHS)

FHS is the prospective cohort study started in 1948 and continuing up to this day. The original objective of the Framingham Heart Study was to identify the risk factors that contribute to CVD developing. The original study cohort consisted of 5 209 respondents of a random sample of 2/3 of adults, 30 to 62 years of age, residing in Framingham, Massachusetts, USA in 1948. The Offspring Study was initiated in 1971 when the need for establishing a prospective epidemiologic study of young adults was recognized. A sample of 5 135 men and women, consisting of the offspring of the original cohort and their spouses, was established, see `http://www.nhlbi.nih.gov/about/framingham/design.htm`.

In 1991, the Framingham CHD risk function was derived from 2 590 men at the age of 30 to 74 years, who were free of cardiovascular disease (stroke, transient ischemia, CHD, congestive heart failure and intermittent claudication) at the time of examinations in 1971-1974 [1]. The Framingham function of

age [years],
systolic blood pressure (SBD – average of two office measurements) [mm Hg],
cholesterol (total serum cholesterol) [mg/dl],
high density lipoprotein cholesterol (HDL) [mg/dl],
smoking (1, cigarette smoking or quit within past year; 0, otherwise),
diabetes (1, diabetes; 0, otherwise) and
electrocardiography – left ventricular hypertrophy (ECG LVH) (1, definite; 0, otherwise)

was estimated to predict CHD developing within 4–12 years. There are some differences in the equation calculation of CHD risk for men and women. For men, the predicted probability (p) of CHD within t years is

$$p = 1 - \exp\left(-e^{u}\right), \tag{1}$$

where

$$u = \frac{\log(t) - \mu}{\sigma},$$
$$\sigma = \exp\left(-0.3155 - 0.2784 \cdot m\right),$$
$$\mu = 4.41818 + m,$$
$$m = a - 1.4792 \cdot \log(\text{age}) - 0.1759 \cdot \text{diabetes}\ ,$$
$$a = 11.1122 - 0.9119 \cdot \log(\text{SBP}) - 0.2767 \cdot \text{smoking} - 0.7181 \cdot \log(\text{cholesterol}/\text{HDL}) - 0.5864 \cdot \text{ECG LVH}.$$

Statistical methods

In STULONG, the Framingham CHD risk was estimated according to (1), on the assumption that HDL is equal to 38.66 mg/dl (the level of HDL was not ascertained at the entry) . The accuracy of risk to predict CHD within 10-year period was evaluated by the Receiver Operating Characteristic (ROC) curve and chi-square goodness of the fit test.

3. Results

Out of 1 248 men from the normal and the risk groups, 123 men without information on all risk factors at the entry into the study were excluded from the statistical analysis. The statistical analysis did not involve the men from the pathological group either.

For 1 125 out of 1 248 men from the normal and the risk groups, the characteristics of risk factors at the entry into the study are shown in Table 2. At the entry into the study, none of 1 125 men suffered from diabetes mellitus and left ventricular hypertrophy, 53 % of men were smokers.

Table 2: Age, systolic blood pressure (SBD), diastolic blood pressure (DBD) and total cholesterol (TCH) of men at the entry into the study ($n = 1\,125$)

Variable	Mean	Std.Dev.	Min	Max
Age	46.1	3.6	38.0	53.0
SBP	131.6	18.0	90.0	210.0
DBP	83.5	11.5	50.0	135.0
TCH	6.0	1.2	2.9	12.2

The men were divided into four categories according to the value of the Framingham coronary heart disease risk, see Table 3. During 10 years from the entry into the study, 141 CHD were predicted, and 116 observed. No significant difference was found between the expected and observed numbers. The cumulative number of the observed CHD is shown on Figure 2.

According to the ROC curve, the Framingham CHD risk classified the men free of CHD at the entry into the study into those with and without CHD within 10 years with 74 % accuracy, see Figure 3 – the area under the ROC curve is representing 74 %. The sensitivity of the diagnostic test (i.e. the Framingham CHD risk >14 % at the entry into the study) was 69 %, the specificity 68 %.

Table 3: Number of men (n) and the observed and expected numbers of coronary heart diseases (CHD) during 10 years from the entry into the study

Framingham risk at the entry into the study	n	Observed CHD (%)	Expected CHD (%)
< 5	58	0.0	4.1
<5, 10)	349	4.0	7.8
<10, 15)	377	8.0	12.4
<15, 20)	244	16.8	17.2
≥ 20	97	32.0	23.6
Total	1 125	10.3	12.5

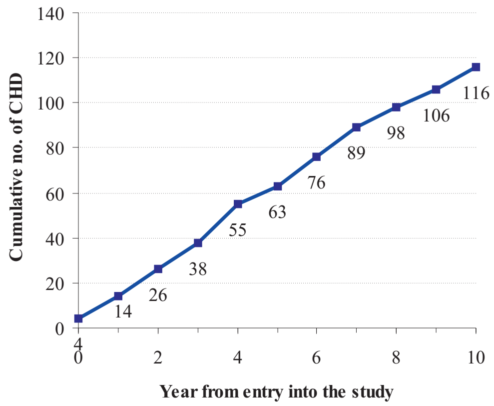


Figure 2: The cumulative number of heart coronary heart disease (CHD)

4. Discussion

Validation of the Framingham coronary heart disease prediction is the aim of epidemiological studies. The Framingham CHD prediction functions perform well among whites and blacks in different settings and can be applied to other ethnic groups after recalibration for differing prevalences of risk factors and underlying rates of CHD events [4].

The validity of external predictive models is assessing [4] by

- comparison relative risk factors between external and studied populations,
- discrimination, and
- calibration.

The aim of the work was to validate the Framingham risk function derived from FHS in STULONG, i.e. in men of the Czech Republic. We used the methods of discrimination and calibration to validate the Framingham coronary heart disease prediction.

Discrimination is the ability of a prediction model to separate the group being tested into those with and without the disease in question. The ability of the Framingham CHD risk prediction to discriminate between men who developed CHD within 10-year from the entry into the study and those who did not was good (74 %). The values of the sensitivity and the specificity of the diagnostic test (i.e. the Framingham CHD risk) were comparable, when the Framingham CHD risk of 14 % was chosen as the bordeline. The sensitivity

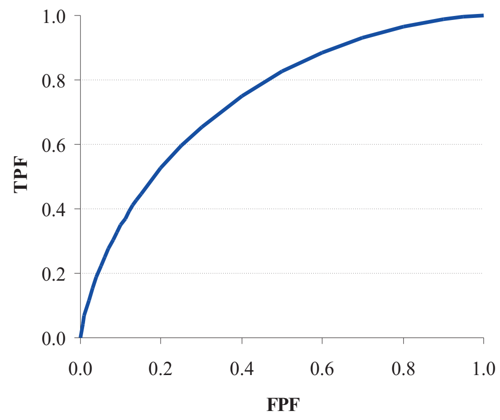


Figure 3: Receiver Operating Characteristic (ROC) curve (FPF – False Positive Fraction, TPF – True Positive Fraction)

of 69 % was the proportion of men with CHD in 10–year follow-up who test positive, i.e. the Framingham CHD risk >14 % at the entry into the study. The specificity of 68 % was the proportion of men without CHD in 10–year follow-up who test negative, i.e. the Framingham CHD risk $\leq$14 % at the entry.

Calibration measures how closely the expected number of disease in question agrees with the observed number. In STULONG, 141 CHD were predicted, 116 observed. No significant difference was found between the expected and observed numbers.

The further goals

The further goals of our work are

- to identify subjects in STULONG whose the Framignham estimate of absolute risk failed,
- to develop adjustments in the FHS risk equation for population the Czech Republic, and
- to validate other predictive models derived from European population.

Conclusion

The work suggested that the Framingham risk seemed to be a fair indicator of a coronary heart disease in men from STULONG.

The study was supported by the project LN00B107 of the Ministry of Education of CR.

References

[1] K. M. Anderson, P. W. Wilson, P. M. Odell, W. B. Kannel, "An Updated Coronary Risk Profile, A Statement for Health Professionals", *Circulation*, vol. 83, pp. 356–362, 1991.

[2] G. Assmann, P. Cullen, H. Schulte, "Simple Scoring Scheme for Calculating the Risk of Acute Coronary Events Based on the 10-year Follow-up of the Prospective Cardiovascular Munster (PROCAM) Study", *Circulation*, vol. 105, pp. 310-315, 2002.

[3] R. B. D'Agostino, M. W. Russell, D. M. Huse, R. C. Ellison, H. Silbershatz, P. W. Wilson, S. C. Hartz, "Primary and Subsequent Coronary Risk Appraisal: New Results from the Framingham Study", *American Heart Journal*, vol. 139, pp. 272–281, 2000.

[4] R. B. D'Agostino, S. Grundy S, L. M. Sullivan, P. Wilson, "Validation of the Framingham Coronary Heart Disease Prediction Scores: Results of a Multiple Ethnic Groups Investigation", *Journal of the American Medical Association*, vol. 11, pp. 180–187, 2001.

[5] S. M. Grundy , R. B. D'Agostino, L. Mosca, G. L. Burke, P. W. Wilson, D. J. Rader, J. I. Cleeman, E. J. Roccella, J. A. Cutler, L. M. Friedman, "Cardiovascular Risk Assessment Based on US Cohort Studies: Findings from a National Heart, Lung, and Blood Institute Workshop",*Circulation*, vol. 104, pp. 491-496, 2001.

[6] Kolektiv řešitelů (připravila: H. Geizerová), "Národní primárně preventivní multifaktoriální studie srdečních infarktů a akutních mozkových příhod v ČSSR", *Praktický lékař*, vol. 60, pp. 463–467, 1980.

[7] S. J. Pocock, V. McCormack, F. Gueyffier, F. Boutitie, R. H. Fagard, J. P. Boissel, "A Score for Predicting Risk of Death from Cardiovascular Disease in Adults with Raised Blood Pressure, Based on Individual Patient Data from Randomised Controlled Trials", *British Medical Journal*, vol. 323, pp. 75-81, 2001.

[8] J. Reissigová, M. Tomečková, "Intervence rizikových faktorů aterosklerózy a kardiovaskulární úmrtnosti, 20letá primárně preventivní studie z pohledu statistika", *Cor et Vasa*, vol. 45, pp. 249–255, 2003.

[9] S. G. Thompson, S. D. Pyke, D. A. Wood, "Using a Coronary Risk Score for Screening and Intervention in General Practice. British Family Heart Study", *Journal of Cardiovascular*, vol. 3, pp. 301–306, 1996.

[10] T. F. Thomsen , M. Davidsen , H. I. T. Jorgensen , G. Jensen , K. Borch-Johnsen, "A New Method for CHD Prediction and Prevention Based on Regional Risk Scores and Randomized Clinical Trials; PRECARD and the Copenhagen Risk Score", *Journal of Cardiovascular*, vol. 8, pp. 291-297, 2001.

[11] H. Tunstall-Pedoe. "The Dundee Coronary Risk-Disk for Management of Change in Risk Factors", *British Medical Journal*, vol. 303, pp. 744-747, 1991.

Entropy-Based Target Functions for Multilayer Neural Networks

doktorand:
MILAN RYDVAN
Ústav informatiky AV ČR
Pod vodárenskou věži 2,
182 07 Praha 8
Czech Republic
rydvan@cs.cas.cz

školitel:
LADISLAV ANDREJ
Ústav informatiky AV ČR
Pod vodárenskou věži 2,
182 07 Praha 8
Czech Republic
andre@cs.cas.cz

obor studia:
I1 - Teoretická informatika

Abstract

The aim of this paper is to discuss application of alternative, entropy-based target functions on multilayer neural networks, comparing them with frequently used least square error function $E = (y-d)^2$. Cross-entropy function and mutual information function based on Shannon entropy are discussed. Genetic training is used in order to allow use of potentially non-continuous and non-differentiable target functions, such as the Shannon entropy function. The research is still in progress; the article presents preliminary results the proposed methods achieved when applied on the real-life problem of stock price prediction.

1. Introduction

This work deals with the model of *multi-layer neural networks*. The model is widely known; the definition can be found for example in [8]. Multi-layer neural networks employ supervised training, using a finite training set $T = \{(\vec{x}_i, \vec{d}_i)\}$ of pairs of input vectors and desired output vectors. The aim of training is to find such parameters of the network (weights, thresholds) that minimise a *target function* $E(d_{ij}, y_{ij})$, summed over all the output neurons and all the training patterns, where y_{ij} stands for the actual output of the network's j-th output neuron for the i-th training input. Because we will work with networks with a single output neurons, we will, for simplification, omit the indices in the following text and use d and y when speaking about scalars (the network's output for a single training pattern) and $\vec{d}$ and $\vec{y}$ when discussing the vectors of the network's output on the whole training set.

Rummelhart ([8]) proposed least square error function $E = (y - d)^2$ as the target function and it is widely used till today. Its advantages include the fact is that it is simple and natural. The fact that it penalises the distance between the desired and the actual output makes it applicable, with a better or worse success, on all kinds of problems without requiring a specific knowledge about the character of the problem.

The author's previous works have studied alternative target functions based on effort to teach the network some specific knowledge. The basic idea of this approach is that the least square error function is too general and cannot express particular pieces of knowledge we may have about a specific problem. We proposed to use biquadratic target functions [5], relief error networks [6] and genetically trained neural networks allowing for arbitrary target functions [7] in order to express more specific target functions and teach the network the extra knowledge. The results have shown that this method is feasible. Networks trained using the mentioned specific target functions had better results and improved generalisation on a testing problem in comparison with those employing the standard least-square error function.

This article represents a different way for research. The aim is to propose and test alternative target functions which are general in the sense mentioned above, i.e. which do not require specific knowledge on top of the training set. In particular, we focused on entropy-based target functions.

2. Entropy

Entropy is a quantity originating in thermodynamics, describing the measure of disorder in a system. This in another words means that it describes also the measure of information contained within a system. We will use this fact when applying entropy-based functions as target functions for neural networks.

The first of the functions we propose is the *cross-entropy function*:[1]

$$E_c = \left(d \ln \frac{d}{y} + (1-d) \ln \frac{1-d}{1-y} \right), \qquad d, y \in (0,1). \tag{1}$$

In order to be able to use this function as a target function for neural networks trained by the Back-Propagation training algorithm, we need to compute its derivative according to y:

$$\frac{\partial E_c}{\partial y} = d \cdot \frac{y}{d} \cdot \frac{-d}{y^2} + (1-d) \cdot \frac{1-y}{1-d} \cdot \frac{1-d}{(1-d)^2} = \frac{1-d}{1-y} - \frac{d}{y}. \tag{2}$$

We can see that $E_c = 0$ if and only if $d = y$; the minimum of E is located in these points. For the graph of E_c, see Figure 1.

The other target function we will propose in this article is based on the *Shannon entropy*. We will define this measure using probability distribution (for more information, see for example [2]). Let us have an observable x with the probability density $d\mu/dx$. Let us cover the space of the prospective values of X with B disjoint bins $\mathcal{P}_j$ of equal side length. Let

$$p_j = \int_{\mathcal{P}_j} d\mu(x) \tag{3}$$

be the probability that x falls into the j-th bin. The Shannon entropy of the observable x is then defined as

$$H(x) = - \sum_{j; p(j) \neq 0} p_j \ln p_j. \tag{4}$$

Similarly, we define *joint Shannon entropy* of two observables, x and y, with probability densities $d\mu_x/dx$, $d\mu_y/dy$. The number of bins is the square of the number of bins used for the observables separately, as each

[1] In fact, we use the negative value of this function, because we will minimise it.

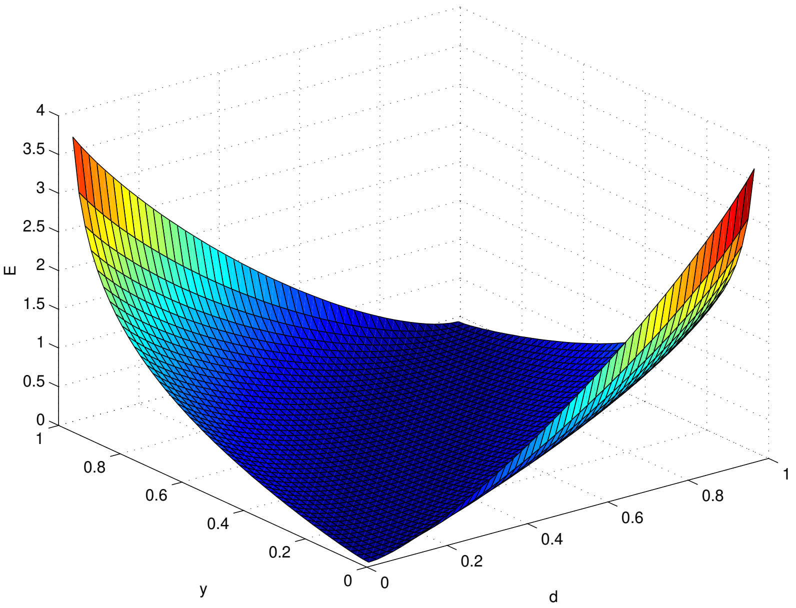


Figure 1: The cross-entropy target function

combination is examined. Let

$$p_{ij} = \int_{\mathcal{P}_i \mathcal{P}_j} d\mu(x) d\mu(y) \tag{5}$$

denote the probability that x falls into the i-th bin and y falls into the j-th bin. The joint Shannon entropy of observables x, y is then defined as

$$H(x, y) = -\sum_{ij; p_{ij} \neq 0} p_{ij} \ln p_{ij}. \tag{6}$$

Note that:

$$y = d \Rightarrow H(x) = H(y) = H(x, y), \tag{7}$$

because only the diagonal bins ($i = j$) are non-empty in the joint Shannon entropy and p_{ii} defined in the joint entropy equals p_i from the individual entropies for every i. In the opposite case, when both x and y represent uniform random distribution, i.e. they are completely independent on each other,

$$x, y \in U \Rightarrow H(x, y) = -\sum_{1}^{B^2} \frac{1}{B^2} \ln \frac{1}{B^2} = 2lnB = -2\sum_{1}^{B} \frac{1}{B} \ln \frac{1}{B} = 2H(x) = 2H(y). \tag{8}$$

The target function we will use is the inverted value[2] of the *normalised mutual information* function, and it

[2]Because we will use the function as fitness function for genetic algorithms and therefore maximise it.

is defined as follows.

$$E_S(\vec{d}, \vec{y}) = \frac{H(\vec{d}) + H(\vec{y})}{H(\vec{d}, \vec{y})}. \tag{9}$$

Equations (7) and (8) imply that $E_S(\vec{d}, \vec{y})$ equals 2 for $\vec{d} = \vec{y}$ and limits to 1 for independent $\vec{d}$ and $\vec{y}$, represented by random noise with uniform distribution. Our aim therefore is to maximise E_S. We however cannot employ the Back-Propagation algorithm because from definitions (4), (6), (9), neither $H(x)$ nor $H(x, y)$ have derivative in y, and therefore E_S does not have it, either. We must thus employ an alternative teaching method; genetic training was chosen for its general character — it enables use of any type of target function.

3. BP-Training

Thanks to the fact that cross-entropy target function (1) has a derivative in y, we may employ the Back-Propagation algorithm for its training. For detailed information about this algorithm, see e.g. [8]. Let us just remind that it is based on adaptation of the neural networks' parameters as follows:

$$w^{\text{new}} = w^{\text{old}} - \alpha \frac{\partial E}{\partial w^{\text{old}}}, \tag{10}$$

where α is the *learning rate*. It represents the speed of learning. Target function plays a direct role in the case of an output neuron, where it holds:

$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w_i} \cdot y, \tag{11}$$

The first partial derivative in this equation is derived in (2); the second one, as well as derivation of the weight changes in the hidden neurons, is the same as in the original BP-training algorithm.

4. Genetic Training

Networks using the Shannon-entropy based target function (9) will be trained by means of genetic algorithms. These algorithms (see for example [4] for more detailed information) perform distributed cooperating search in the solution space. Each prospective solution is coded in the form of a chromosome, a string of one-bit, two-bit or real values. Each chromosome is assigned a *fitness*, which measures how suitable the corresponding solution is. The GA maximises the fitness using genetic operators on a population of chromosomes. *Selection* ensures the overall improvement of fitness, *crossover* combines schemes in existing chromosomes in order to create new patterns in new chromosomes and *mutation* makes random modifications, helping the system to produce new prospective solutions and avoid local minima.

When training neural networks using GAs, the chromosome can consist of real-valued genes, each representing a single parameter of the network — a weight or a threshold. The fitness of such chromosome-network is the value of the target function E_S applied on the network and the training set.

Genetic training of neural networks usually has several drawbacks compared to gradient methods — it tends to be slower and its results are poorer. On the other hand, it does not suffer from the local minima problem so much. However, the main benefit of genetic training for us is that it allows usage of general target functions. The aim of the research, whose preliminary results are presented in this work, is to discover whether the benefits from the use of this target function will balance (and hopefully outperform) the drawbacks of genetic training.

5. Stock Price Prediction

We have compared the performance of networks trained using the proposed target functions with those using the least-square error function on the problem of stock price prediction. The aim was to predict the stock price change in the following trading day, knowing a history of (five) previous price changes and additional information about the last trading day, such as the volume of trade, the position of the latest known price in the long-term history, the supply/demand ratio etc. The raw data from the stock exchange were subject to extensive pre-processing. It was for example necessary to transform the outputs of the network, which represent the expected price change, i.e. generally a real number, into the interval $(0, 1)$, because it is required by the target function (1). This was achieved by transformation using the sigmoidal function

$$\bar{y} = f(y) = \frac{1}{1 + e^{-y}}. \tag{12}$$

Among other pre-processing methods there was e.g. application of the Principal Component Analysis (PCA) (see [3]) on the data. The PCA normalises the data and therefore increases the performance of training.

We used Matlab as the platform for programming the experiments. Matlab's Neural Network toolbox was used, together with the author's implementation of alternative target functions. In order to implement genetic training, we have interconnected this toolbox with a GA toolbox developed by Houck, Joines and Kay ([1]). Series of experiments were carried out, in order to determine and tune the parameters of the tested methods; the results are, however still preliminary. We used the same architecture (9-15-1) for both the standard least-square error function and the alternative target functions, in order to keep the conditions of the compared models as similar as possible.

Firstly, let us deal with the cross-entropy target function (1), trained using the Back-Propagation training algorithm. This target function has shown to be very sensitive on the learning rate. The reason is that $\lim_{y\to 0} \frac{\partial E_C}{\partial y} = -\infty$ and $\lim_{y\to 1} \frac{\partial E_C}{\partial y} = \infty$ (see (2)). Values $y = 0$ and $y = 1$ after pre-processing of the data represent infinite slump and growth of the stock price, respectively, (see (12)) and similar extreme values therefore should not appear in a trained network; they may however appear in a fresh network that has been created randomly and has not undergone much training yet. This problem may be solved by applying a very low learning rate α. This however makes the training process very slow, and increases the risk of getting stuck in a (very) local minimum. Therefore, we have chosen a method of variable learning rate. At the beginning of the training, when the chance of extreme values of y is larger, α is low ($2.5 \cdot 10^{-4}$). It is then doubled twice, after 100th and 200th iteration of the Back-Propagation, when it thus reaches $1 \cdot 10^{-3}$. This limits the problem of diverging changes of network parameter to a manageable level.

The training process with the use of target function (1) shows one more, positive, interesting feature. With the exception of the above-mentioned "pathologic" cases of diverging changes of the parameters, the networks tend to achieve very similar results when lower and higher values of learning rate α are used.[3] This suggests that networks using this function are apparently less tending to get stuck in a local minima.

As regards the normalised mutual information target function (9) and genetic training, the networks with architecture 9-15-1 have 150 weights and 16 thresholds and each network was thus represented by a real-value chromosome with 166 genes. For sake of speed, population was limited by the maximum of 50 chromosomes. The computation of the target function E_S is rather slow, as it requires computation of histograms of size B for the individual Shannon entropy and of size B^2 for the joint Shannon entropy. Moreover, a feature of genetic neural network training generally is that in order to compute the fitness of a chromosome (e.g. when applying selection operator), we must apply the neural network on the whole training set. The length of computation of other genetic operators also depends on the number of chromosomes in the population. Therefore, this number had to be kept low.

We used the roulette wheel selection operator, which selects a chromosome with the probability proportional

[3] The training time is indeed different.

to its fitness. The mechanism of elitism was employed so that the best chromosome from the population always survived. Arithmetic crossover was employed. This operator produces two complimentary linear combinations of real-valued parent chromosomes (X, Y):

$$X' = rX + (1 - r)Y \tag{13}$$

$$Y' = (1 - r)X + rY, \tag{14}$$

based on a random value $r \in U(0, 1)$. Uniform mutation randomly selects one gene in one chromosome and assigns it a uniform random number from the permitted space of values (interval $(-10, 10)$ was used as the permitted space for the gene values).[4] The probabilities of crossover and mutation were 0.15 and 0.5, respectively.

In order to estimate and compare generalisation abilities of the methods, we divided the known data into a *training set*, which was used during the training period, and the *test set*, unseen by the networks during training and used for measuring their performance on unknown data. The training set contained 75% of the data; the test set contained the remaining 25%.

We compared the proposed methods by carrying out 100 experiments. During each of them, three networks were trained - one using the standard least-square error function and one using each of the proposed target functions (1) and (9). The table below contains the averaged results both on the training set and the test set. The division into the training/test set was carried out randomly for each of the 100 experiments; it was however the same for all three target functions tested.

Target function	Set	Square error	Direction correctness	Profit
LSE	Train	0.033	81.9%	0.478%
	Test	0.051	72.0%	0.165%
(1)	Train	0.041	76.4%	0.333%
	Test	0.048	72.1%	0.194%
(9)	Train	0.218	70.9%	0.092%
	Test	0.226	67.2%	0.079%

Table 1: Comparison of performance of the least-square error function and of the proposed target functions on the problem of stock price prediction, separately for the training set and the test set. The first and second couple of rows contain the results for networks trained by BP-training algorithm and applying the least-square error function and the cross-entropy function, respectively. The third couple presents the results of networks trained genetically, using the normalised mutual information. Several measures of success are presented - summed-square error, direction correctness (the percentage of correct prediction of price rise/decrease) and an average daily profit model.

As we can see, the two alternative target functions have produced very different results. As regards the outcomes on the tests set, the cross-entropy function (1) has exceeded the least-square error function in all three measures used — the direction correctness, the summed square error itself(!) and, above all, the profit model.

The fact that the results on the test set were better even though the results on the training set were worse than in the case of the standard error function suggests that the generalisation ability of the cross-entropy function is better. The cross-entropy function was also significantly faster (it needed only 273 training cycles in average to achieve these results, in comparison with 953 cycles in the case of the least=square error function).

[4]For detailed definition of the mentioned genetic operators, see [1].

On the other hand, the outcome of the normalised mutual information function (9) are worse, both on the training and the test set. One of the reasons may be the high time complexity of training using this target function; it was therefore not yet possible to test larger populations and tune the parameters of the tested networks and genetic algorithms appropriately.

Let us however stress that these results are not bad either — the direction correctness is close to 70% and the average net (including transaction costs) daily profit (though only modelled) is approaching 0.1% daily. Moreover, we can note that the results of the networks trained using function (9) are very similar for the training set and for the test set. This suggests good generalisation abilities of this target function. The fact that these networks tend to give ”trading advices” (i.e. predict a share price change higher than the transaction costs) more often than those trained using the least-square error is promising as well; the author's previous works ([5],[6],[7]) suggest that ”more courageous” predictions were awarded by higher profit.

6. Conclusion

This article proposed two types of alternative entropy-based target functions for multilayer neural networks. One of them is based on the cross-entropy function and employs standard Back-Propagation training algorithm, the other is normalised mutual information based on Shannon entropy and employs genetic training.

Theoretic derivation of the appropriate functions, their properties and training methods was accompanied by preliminary results of their testing on a real-life sample problem — stock price prediction. The results of the cross-entropy function exceeded those achieved by networks trained by the standard least-square error function; the results of the mutual information function were worse. Both alternative target functions proposed in this paper show better generalisation abilities.

The future work in this field will include further thorough tuning of the systems' parameters. More time is needed above all in the case of the model based on Shannon entropy, whose results are not very good so far, partially due to the very time-consuming character of its training. Further work is also needed to speed up this method. Alternative functions based on these and other entropies offer a wide field for research, and as the results of the cross-entropy function suggest, the outcome may be worth the effort.

References

[1] Chris Houck, Jeff Joines, Mike Kay, *“A Genetic Algorithm for Function Optimization: A Matlab Implementation”*, NCSU-IE TR 95-09, 1995, see `http://www.ie.ncsu.edu/mirage/#GAOT`.

[2] H. Kantz, Thomas Schreiber, *“Nonlinear Time Series Analysis”*, Cambridge University Press, 1997.

[3] T. Masters, *“Advanced Algorithms for Neural Networks”*, John Wiley & Sons, 1995.

[4] M. Mitchell, *“An Introduction to Genetic Algorithms”*, MIT Press, 1997.

[5] M. Rydvan, *“Biquadratic Error Functions for the BP-networks”*, TR No 98/10, Charles University Prague, MFF.

[6] I. Mrázová, M.Rydvan, *“Generalised Error Function for BP-Network”*, TR No 99/1, Charles University Prague, MFF.

[7] M.Rydvan, *“Generalised Target Functions for Multilayer Neural Networks”*, Doktorandský den '02, Institute of Computer Science, Czech Academy of Sciences.

[8] D. E. Rummelhart, G. E. Hilton,R. J. Williams, *“Learning representations by backpropagating errors”*, Nature (323), (1986), pp. 533-536, MFF.

GUHA inspired methods for fuzzy data mining

doktorand:
MARTIN SATURKA
ICS CAS CZ

kvutza@cs.cas.cz

školitel:
PETR HÁJEK
ICS CAS CZ

hajek@cs.cas.cz

obor studia:
Theoretical Computer Science

Abstract

Standard GUHA [3] data mining methods are for crisp, i.e. two-valued {yes, no}, data. We develop algorithms to exploit continuous data in a natural way. The task is motivated by growing amount of biological data that come from microarray experiments. We describe data representations which are suitable for logical transformations based on gene expression regulations. We define multitudinal quantification on fuzzy data. Next to it, we make series of such quantifiers and we describe their features.

1. Introduction

There is growing amount of biological data which are usable for data mining procedures. Microarray experiments are promising source of such data. Result from one microarray experiment is an amount of floating point numbers. The amount is frequently thousands of values. One such value indicates a level of expression of one specific gene (i.e. production level of its coded protein). Thus, data from one experiment describe how particular genes are expressed under specific experimental conditions. One set of experimental conditions can be a specific tissue, cancer cells, etc. Collection of values from several experiments can be viewed as data matrix where objects (rows) are experiments and attributes (columns) are genes. Data from one row are usually called an expression profile. The challenge is exploration of relations between expression levels of particular genes. The goal of this work is to provide methods which are suitable for data mining on the data collections introduced above.

Genes (that are four-symbol sequences along DNA molecules) code protein sequences. Protein production according to DNA sequences is called gene expression. The expression process is mediated by mRNA molecules. Amounts of mRNAs of particular genes determine levels of their expressions and thus regulations. There are developed technics to extract, label and identify mRNAs from biological materials (e.g. microarray experiments) [2]. Experimental results (i.e. expression values) are gained primarily as light signals. There are two main classes for the light signals. First, the signals can directly reflect expression levels under investigated conditions. Second, the signals can reflect ratios between expression levels under investigated and reference conditions. The first possibility suffers from the fact that absolute values of the light signals depend on variable values of experiment parameters. Ratios between expression levels of individual genes are the relevant data. The second possibility suffers from unknown exact expression levels under reference conditions.

Number of researchers design and develop methods to extract information from microarray experiments data. There are standard methods to transform and normalize the data [5]. Data normalization reflects the fact that positive and negative regulations contain information of the same importance. Thus, values of the same greatness should be taken from expression values which are either k-times higher than or k-times lower than basic expression values. It is yielded by logarithming the data. There should be performed some divisions of data by chosen standard values before doing the logarithm. Mean (or another base) value of data from one experiment is an essential divisor in case of experiments without paired mRNAs. The most common current technics for data mining itself comprise gene clustering and expression profile comparison. Clustering methods are used to find similarly regulated genes. Profile comparisons are used for diagnosis (i.e. classification) purposes.

2. Data representation

Interval $[-1, 1]$ is chosen as a natural way to express negative regulation (i.e. interval $[-1, 0)$), basic value (i.e. 0 value), and positive regulation (i.e. interval $(0, 1]$). Higher values are for greater levels of gene expressions, lower values are for lesser levels of gene expressions. We write terms in scope of propositional logic language in this section.

There can be found four classes of mappings which play roles of conjunctions and disjunctions. First, let us evaluate proposition which says that both gene a and gene b are activated. Result value of this conjunction should be (may nonstrictly) lower than input values. Second, let us evaluate proposition which says that at least one of genes a and b is activated. Result value of this disjunction should be (may nonstrictly) higher than input values. Situation is opposite to the above if we want to make formulas on gene inhibitions. We need to do formulas on combinations of both activations and inhibitions as well. General solution of this situation is the usage of 2-tuples that are to be assigned to formulas. Thus, each formula has both its evaluation and its assigned 2-tuple. Every 2-tuple has both its values at interval $[0, 1]$. There has to be one more operation, say ”opposite”, which has one argument. This operation exchanges the first and the second 2-tuple values of its argument. The operation has a role which somewhat resembles negation. Doubling the opposite results into the initial formula. Evaluation of a formula is the first value of its 2-tuple. The second value of the 2-tuple is the evaluation of a formula with the opposite meaning.

It is obvious that ”opposite of (a and b)” should be equivalent to ”(opposite of a) and (opposite of b)” if we want to preserve natural meaning of the connectives. In other words: ”inhibition of both genes a and b” means ”inhibition of gene a and inhibition of gene b”. All the other classical connectives can be used in the same way. Thus, the opposite is, say, outer operation. All the other operations are, say, inner operations.

Let us do an example to make it more clear. Consider formulas on single genes a and b, with assigned 2-tuple $\{a_1, a_2\}$, $\{b_1, b_2\}$. The first values (i.e. a_1, b_1) mean activation levels of the genes a, b. The second values (i.e. a_2, b_2) mean inhibition levels of the genes a, b. Formula of ”gene a is activated” has assigned 2-tuple $\{a_1, a_2\}$ and is evaluated as a_1. More complex formula is, for example, ”both genes a and b are activated”. It has assigned 2-tuple $\{a_1 \text{ and } b_1, a_2 \text{ and } b_2\}$ and its evaluation is a_1 and b_1. The 2-tuple $\{a_2, a_1\}$ expresses regulation values which are opposite to that of gene a. It can be used in construction of formulas like ”gene a is inhibited or gene b is activated”. Its assigned 2-tuple is $\{a_2 \text{ or } b_1, a_1 \text{ or } b_2\}$ and its evaluation is a_2 or b_1.

Taken it all together, correct formulas are basic ones and composite ones. Basic formulas are of form ”object o is activated”, their assigned 2-tuples are $\{o_1, o_2\}$ and their evaluations are o_1. Composite formulas are made from other formulas by the ”outer” operation of opposite and ”inner” operations conjunction, disjunction and possibly other ones like implication, coimplication and negation. Formula created by the opposite from ϕ, $(\{\phi_1, \phi_2\}, \phi_1)$ is of form $-\phi$, i.e. ”opposite of ϕ”. It has assigned 2-tuple $\{\phi_2, \phi_1\}$ and an evaluation ϕ_2. Formula created by an inner binary (unary respectively) operation op from ϕ, ψ formulas,

(ϕ formula respectively), is of form op(ϕ,ψ), (op(ϕ) respectively). Its assigned 2-tuple is of form $\{$op(ϕ_1,ψ_1), op(ϕ_2,ψ_2)$\}$, ($\{$op(ϕ_1), op(ϕ_2)$\}$ respectively).

Up to now, we have considered formulas as abstract ones with assigned 2-tuples. But there is a question: How to obtain the 2-tuple assignment for basic formulas. We have to take two values which fit in interval $[0, 1]$ from one value (label it by "v") that fits in interval $[-1, 1]$. Lower (higher respectively) is the initial value, lesser (greater respectively) should be the first value of the pair and greater (lesser respectively) should be the second value of the pair. Let us consider there is some basic (label it by z) value (zero value was considered in the beginning of this section) which means there is no activation and no inhibition. The 2-tuple is $\{$distance(v,z), $0\}$ in case of high (positive) values, $\{0,$ distance(v,z)$\}$ in case of low (negative) values, and $\{0, 0\}$ in case of equality of v and z.

It can be generalized as shown at Figure 1. The intervals for activation and inhibition values can be in different relative positions. They can, for example, overlap or have nonzero distance. It can be described by a set of connected relations. We assume some symmetry conditions, since there is no prerequisite need for any asymmetry. The class of models can be, of course, directly generalized to contain asymmetrical situations.

First, we have output of double use of the opposite operation equal to the input. In terms it is $-(-\phi)_i = \phi_i$, $i \in \{1, 2\}$ for assigned 2-tuple values and ϕ formula, or without equality operation $\phi \rightarrow -(-\phi)$ and $-(-\phi) \rightarrow \phi$ for ϕ formula. These ones hold for every formula. We have more holding relations when we restrict ourselves to basic formulas and their opposites. Let us assume a set of new formulas I_α that are labeled by parameter $\alpha \in [0, 1]$. Evaluation of each formula I_α is α, the second values of their assigned 2-tuples can be arbitrary, e.g. zero. Then we have two sets of formula pairs ϕ_i, ψ_i, which means: when we have proved ϕ_i, we can deduce ψ_i. Formulas in the first set are $\phi_i \equiv \mathrm{I}_\alpha \rightarrow a$ and $\psi_i \equiv -a \rightarrow \mathrm{I}_\beta$ where a is any basic formula or its n-times repeated opposite. Formulas in the second set are $\phi_i \equiv a \rightarrow \mathrm{I}_\gamma$ and $\psi_i \equiv \mathrm{I}_\delta \rightarrow -a$ where a is again any basic formula or its n-times repeated opposite. Distinct classes of models differ in values for α, β, γ, and δ which can be used for correct deduction pairs.

First, let us consider models which fit to situations A, B at Figure 1. Then the valid values are each $\beta \in [0, 1]$ with each $\alpha \in (0, 1]$, $\beta = 1$ with $\alpha = 0$, $\delta = 0$ with each $\gamma \in [0, 1]$. Second, let us consider models which fit to situation C. Then the valid values are each $\beta \in [0, 1]$ with each $\alpha \in [\mathrm{n0}, 1]$, each $\beta \in [\mathrm{p0}, 1]$ with each $\alpha \in (0, \mathrm{n0})$, $\beta = 1$ with $\alpha = 0$, each $\delta \in [\mathrm{p0}, 1]$ with $\gamma = 0$, $\delta = 0$ with each $\gamma \in (0, 1]$. We set $\mathrm{p0} = \mathrm{n0}$ for the purpose of symmetry. Third, let us consider models which fit to situation D. Then the valid values are each $\beta \in [0, 1]$ with $\alpha = 1$, $\beta = 1$ with each $\alpha \in (0, 1]$, each $\delta \in [0, 1]$ with $\gamma = 0$, $\delta = 0$ with each $\gamma \in (0, 1]$. The other situation cases are somewhat unnatural from the point of regulatory view - positive regulation values are lesser than negative regulation values. However, there can be some other needs for such models. Fourth, let us consider models which fit to situation E. Then the valid values are each $\beta \in [\mathrm{p1}, 1]$ with $\alpha = 1$, $\beta = 1$ with each $\alpha \in [0, 1)$, each $\delta \in [0, 1]$ with each $\gamma \in [0, \mathrm{n1}]$, each $\delta \in [0, \mathrm{p1}]$ with each $\gamma \in (\mathrm{n1}, 1)$, $\delta = 0$ with $\gamma = 1$. We again set symmetry condition $\mathrm{p1} = \mathrm{n1}$. Fifth, let us consider models which fit to situations F, G. Then the valid values are $\beta = 1$ with each $\alpha \in [0, 1]$, each $\delta = [0, 1]$ with each $\gamma \in [0, 1)$, $\delta = 0$ with $\gamma = 1$. It should be noted that the formula pairs are not valid for arbitrary composite formula in place of formula positions a. Let us set the model B as the default model.

There is nothing new from logical point of view when we restrict ourselves to inner operations. In such a case, the 2-tuples can be neglected and we have the class of standard fuzzy logics. The operation of opposite brings new formulas and their evaluations. The formula pairs stated above can be used as a deduction extension. One another extension which is inspired by the above one, uses the outer relation as negation, not the opposite. In such a case, there is a connection between inner operations: conjunction is paired with disjunction and implication is paired with coimplication. The pairing involves assigned 2-tuples making. Assigned 2-tuple of a formula of conjunction (ϕ, ψ) is $\{$conjunction (ϕ, ψ), disjunction (ϕ, ψ)$\}$ and reversely for disjunction. Assigned 2-tuple of a formula of implication (ϕ, ψ) is $\{$implication (ϕ, ψ), coimplication (ϕ, ψ)$\}$ and reversely for coimplication. There is again the need for two values for each basic formula. However, the second values are negations and not opposites in this case. This results in the same outputs as classical predicate calculus when we restrict ourselves into crisp, i.e.$\{0, 1\}$, values with

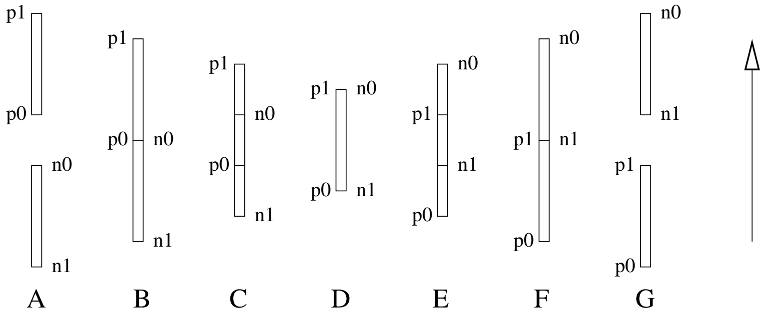


Figure 1: Resection of initial $[-1,1]$ valuation interval into two $[0,1]$ intervals. Positive regulation ranges from $p0$ to $p1$, negative regulation ranges from $n0$ to $n1$. Arrow points to greater expression values.

classical negations for assigned 2-tuples of basic formulas. For example, "outer negation (disjunction (outer negation (ϕ), outer negation (ψ)))" is equivalent to "conjunction (ϕ, ψ)". In the next, we use symbol of x for evaluation value of ϕ, symbol of y for evaluation value of ψ.

After having structure for formula building, it is necessary to choose evaluation rules for inner operations. Inner operations which are needful for data mining, are conjunction and disjunction. The initial model building bids $\min\{1, \log_r(r^x + r^y - 1)\}$, $r > 1$ (label it s_r), where r is the base used for logarithm in data normalization. Such operation is archimedean nilpotent t-conorm with $s_r(0.5, 0.5) < 1$. It moves its input values back to the original interval (before logarithming), carries out Lukasiewicz conorm and then it brings result back. Limits of s_r conorms with $r_+ \to 1$, $r \to \infty$ respectively, are Lukasiewicz, maximum respectively, conorms. The s_r conorms can be defined for $r \in (0,1)$ too. Note that the latter s_r cases need to have set a convention of $\log_r t = \infty$ for $r \in (0,1)$ and $t \leq 0$. Such s_r conorms are again nilpotent archimedean ones. Limits of the s_r conorms with $r_- \to 1, r_+ \to 0$ respectively, are Lukasiewicz, drastic respectively, conorms. There is standard way to make t-norms (i.e. "and" connectives) from conorms (i.e. "or" connectives) and vice versa. It uses standard negation $N = 1 - x$; in fact, it can use any strong negation. It is $T(x, y) = N(S(N(x), N(y)))$ where T is t-norm, S is conorm and N is the standard negation. Label the result t-norm with t_r. Unfortunately, t_r and s_r enumerations are computation intensive for nonlimit cases. There are standard operations which are usually used for conjunction and disjunction computations. They are minimum and maximum, their enumerations are fast - it is necessary in high volume number crunching.

The outer operation can be gained by operations which extend implication-like operations beyond zero ground. Result of the opposite on x is given as output value of "extended operation (x, 0)". It is notable that it is similar to gaining the negation from residual implication. There is more than one possibility how to do it. It can be done by changing the implication from Lukasiewicz logic by setting the neutral element to zero. The result $0 + y - x$ is gained from original $\min\{1, 1 + y - x)$. More direct way is the usage of coimplication which belongs to Lukasiewicz conorm. Then we just omit maximization, i.e. we have $y - x$ from original $\max\{0, y - x\}$. One more way is the usage of mean-implication. As implications are gained from norms, i.e. operations which downsize values, and coimplications are gained from conorms, i.e. operations which upsize values, mean-implications are gained (by an appropriate definition) from means, i.e. operations which makes averages. In such a case, we can use arithmetic mean as the mean operation. Original mean-implication $\min\{0, \max\{1, 2y - x\}$ is extended to $2y - x$.

Some points herein evoke notion of uninorms [1]. However, there are used 2-tuples for formula evaluations in the situation which is described above. It is not available in uninorms. We can restrict ourselves to the default model (see figure 1, situation B) and to operations of the opposite and conjunction. Then we have model where one value of each 2-tuple is zero. In such a case, we can move the model back to single values which are from interval $[-1, 1]$. It is not possible if we want to use either disjunction or model with overlapping intervals. As a result, we have functions which are dual to uninorms, they are nullnorms. While uninorms move values to border values, nullnorms move values to the central value. From this point of view, conjunctions of general model with assigned 2-tuples can be called dinorms.

While implications together with norms (co-implications together with co-norms respectively) can be directly used for proofs of one (zero respectively) validity of formulas, mean-implications have more difficult utilization. Binary averaging operations which are non-decreasing in both arguments, and which have their outputs strictly between input values whenever the inputs are not equal, are not associative. It hardens their utilizations. Nevertheless, we can use them combined with mean-implications without direct nesting the means. Formulas which are made purely from mean-implications, variables and border constants, say 0 and 1, have simple evaluations. Select the rightmost symbol which is either variable or border constant, from the formula. If the selected symbol is a border constant, evaluation is always equal to the constant. Evaluation contains both border constants whenever the selected symbol is a variable. In addition, evaluation results form the hole $[0, 1]$ interval if the used mean-implication is continuous. For example, it holds in the case of arithmetic mean, i.e. $(x + y)/2$ and its mean-implication, i.e. $\min\{0, \max\{1, 2y - x\}$. One example of a formula where both mean operation (label it mo) and its mean-implication (label it mi) are used: mi (x, mo(x, 1)). Its evaluation is 1 for every mean operation (as introduced above) and its residual implication.

3. Multitudination

The previous section was devoted to propositional kinds of logic. Now, we have to consider predicate calculus. Data mining makes heavy utilization of generalized quantifiers. While there are situations where general predicate calculus is needed, we restrict ourselves to unary predicate calculus. It means that we make use of single database tables, and properties are described in terms of single objects (i.e. single rows in database tables). The situation is still fruitful enough. General characterization of data features is done by quantifiers. We usually want to express that most of data objects contains some property. Classical quantifiers express situations when every objects contains or one object contains the property. It is not sufficient in the field of data mining. Thus, generalized quantifiers are used, e.g. 90% of objects contain some property. Next thing we want to do, is to characterize relations between some properties. For example, there is minority of objects which have just one of two given properties (it expresses associativity). Such (binary) quantifiers are over pairs of formulas. The former case can be gained from the latter one by usage of one property and one constant as quantifier inputs. There are two main classes of binary generalized quantifiers: associative and multitudinal ones. Associativity can be vied as function characterization from both sides of its valuation graph. Clustering technics can be viewed as applying some generalized associative quantifiers on input (fuzzy) data. There is amount of such technics. Note that we take interest in fuzzy quantifiers [4], i.e. quantifiers with fuzzy output.

Multitudinal quantifiers express situations where high level of one property assures high level of another property. It can be partly viewed as kind of regression where we count just errors which are caused by cases which lie below some regression curve. Together with it, we do not lay stress on parts of the regression line where there can not be error cases at all. Another difference is that we do not fit curves but we search for fitting compared variables. Let us have a look from gene regulation point of view. Natural interpretation is such that a level of occurrence of some combination of gene regulations guarantees (i.e. it is sufficient but not necessary condition for) the same level of another combination of gene regulations. The former combination is described by formula called antecedent, the latter one is described by formula called succedent. Such properties are easy to describe in the area of crisp data values. In such a case, there are used fourfold tables with values a, b, c, and d. Value of a is count of cases where both antecedent and succedent occur, value of b is count of cases where antecedent occurs but not so succedent. The last two values are for the other cases. A feature is multitudinal if it is diminished neither with increase of "a" value nor with decrease of "b" value. There are developed multitudinal quantifiers over formulas on crisp data, e.g. $a/(a + b)$, but there is a desert field in the case of formulas on fuzzy data. We try to cultivate the field.

Label general antecedent formula with ϕ, its evaluation fuzzy value with x, and general succedent formula with ψ, its evaluation fuzzy value with y. It should be noted that the word of fuzzy does not mean that the data are somehow unknown. It just means that the data can range from 0 to 1. In fact, they can be from another nonempty interval. Then the symbol of 1 is used for maximum and the symbol of 0 is used for minimum. There can be done many generalizations of the crisp multitudinality into the fuzzy area. It is caused by the fact that the value of a (b respectively) in classical fourfold table can be gained as satisfaction (nonsatisfaction respectively) of $\phi \rightarrow \psi$ under condition of ϕ, ψ under condition of ϕ, $\phi\&\psi$ under condition of ϕ, and many other different ones. In all the cases, there is no relevance of values which are under condition where ϕ does not hold. Models under situation when we have fuzzy data, can be more easily understood with 3-dimensional valuation diagrams inside $[0,1]^3$ boxes. They are replacement for fourfold tables. We use 2-dimensional diagrams with valuation contours because of their simplicity.

Figure 2 shows schemas of three basic situations. The model A reflects situation where wrong cases are those with reached level of ϕ not accompanied with the level of ψ. The situation respects implication $\phi \rightarrow \psi$. The model B reflects situation where ψ does not depend on ϕ. The situation respects gation of ψ. The model C reflects situation where we need both ϕ and ψ. The situation respects conjunction $\phi\&\psi$. Each case which is taken from data, falls somewhere onto the square of a chosen model. It takes its evaluation in this way. Occurrences from fourfold tables (i.e. crisp data models) fall onto corners of a chosen square. Model A respects the required assurance of formula levels without any addition. The other models are unnecessarily severe. In fact, model B respects search for ψs which do not depend on ϕs. Model C requires high levels of ψs even under low levels of ϕs. It has no support in initial requirements. Thus model A fills best the demanded conditions. It should be noted that the three shown models are just single representants of classes of models which have the depicted properties. The three models are chosen because of their continuity and linearity.

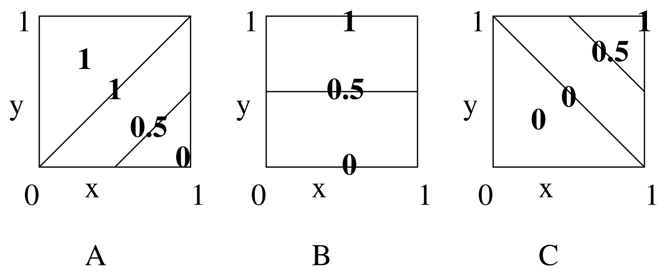


Figure 2: Model (evaluation) squares for formula pairs. The shown cases are representants of three general classes of models.

Quantifiers have to use all the evaluations in a given model set and create an output value. We have to put higher stress on cases which fall more right onto the model square, to retain multitudinal character. In fact, the cases where ϕ have evaluation 1, should have the greatest importance (i.e. weight). Let us scale the maximum onto value 1. Cases where ϕ has evaluation 0, should have zero level weight, i.e. should be neglected. Weight of any case should not be involved by evaluation of ψ. Such conditions are both natural and in concordance with multitudinality on crisp data. Thus, weight function on cases can be expressed as function with one input, i.e. x - the evaluation of ϕ, and one output, the weight. The function is from $[0,1]$ into $[0,1]$. Label the function with F (for force). Then $F(0) = 0$, $F(1) = 1$, and F should be continuous and nondecreasing, since discontinuity points or decrease areas are both unnatural and without reasoning. We can have various weighting functions of such kind. Some of them are shown at Figure 3. The model A shows two classes: less than linearily proportional stress on lower ϕ levels (below $\{0,0\} - \{1,1\}$ diagonal) and greater than linearily proportional stress on lower ϕ levels (above $\{0,0\} - \{1,1\}$ diagonal). They can be such as $F(x) = x^2$, $F(x) = 1 - \surd(1 - x^2)$ for the lowers, $F(x) = \surd x$, $F(x) = \surd(1 - (1 - x)^2)$ for the uppers. The model B shows linear (in parts) weights, for example, the most simple one $F(x) = x$. The model C shows cases where lower ϕ levels tend to be neglected and higher ϕ levels tend to be taken nearly the same as maximal ones. It can be $F(x) = 3x^2 - 2x^3$.

Now, we can put down general definitions of two kinds of fuzzy multitudinal quantifiers on fuzzy data. We can have weak and strong multitudinal quantifiers. Weak fuzzy multitudinal quantifiers on fuzzy data are such mappings from $\bigcup_{n=1}^{\infty}[0,1]^{n\times 2}$ into $[0,1]$ that their restrictions on crisp data are multitudinal quantifier with respect to the standard definition on fourfold tables. Strong fuzzy multitudinal quantifiers on fuzzy data are mappings from $\bigcup_{n=1}^{\infty}[0,1]^{n\times 2}$ into $[0,1]$ which satisfy several conditions. First, the quantifiers are nondecreasing in values at the second positions of pairs that lies inside input n-tuples. Second, change of any pair (inside n-tuple input) which has its values $\{1, 0\}$, into a pair with any other values does not

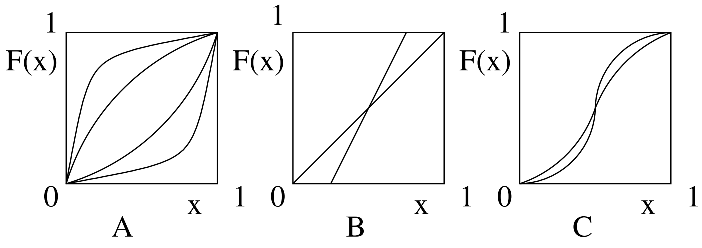


Figure 3: Several examples of weight F functions.

decrease quantifier output. Third, change of any pair (inside n-tuple input) into a pair with its values $\{1, 1\}$ does not decrease quantifier output. Fourth, the quantifiers are invariant to addition and removal of any pair which has zero value in its first position, to/from any input n-tuple whenever $n \neq 0$. Fifth, the quantifiers are invariant to permutations of pairs inside input n-tuples.

Such definitions are somewhat weak because they allow, for example, mappings with constant output. It holds in the case of multitudinal quantifiers on crisp data too. We should realize that usage of shown squares for direct output computations is just one possibility. Quantifiers which fulfill natural requirements for multitudinality, can be more complex. In fact, we do not know apriory how complex they can be and what can be the ways for their evaluations. The upcoming task is to find multitudinal quantifiers which do not have unwanted features. The definitions have some rational properties too, e.g. each strong multitudinal quantifier is weak multitudinal quantifier. It can be viewed from changes of $\{1, 0\}$, $\{1, 1\}$ pairs and additions and removals of $\{0, y\}$ pairs. While general definitions allow usage of all the three models introduced above (see Figure 2), we confine ourselves to situations which are similar to the model A. They are generally model squares of implications (with weights). There are three basic implications which are main representants of such models. They are residual implications of continuous t-norms: Lukasiewicz, product and Goedel ones. Valuation squares with contour lines are good aids to understand particular cases. The squares are shown at Figure 4.

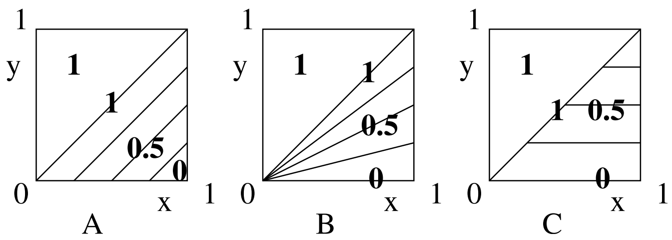


Figure 4: Model squares for three basic implications.

The case A is the evaluation square for Lukasiewicz implication, the case B is the evaluation square for product implication, and the case C is the evaluation square for Goedel implication. The Lukasiewicz implication is continuous, product implication is not continuous at $\{0, 0\}$ point, Goedel implication is not continuous at $\{x, x\}$ points. Continuity is important in case of search for natural relations because nature generally behaves continuously. Since the implication evaluations are just parts of quantifier computations, the discontinuity points can possibly be overcome. It is notable that product implication separates the lower right triangle in equal ratios of higher and lower values. It is preserved after usage of any weighting F function. Lukasiewicz implication results in greater regions of higher values, Goedel implication do it reversely. It is partially attenuated by usage of weighting F functions. We describe several classes of multitudinal quantifiers below.

Weighted (by F functions) arithmetical means of implication evaluations are multitudinal quantifiers. The weights are gained by forces described above. Label implication evaluation with $i(x, y)$, the quantifiers with

$Q_{wm,i,F}$, (wm for weighted mean, F for the used force (weighting) function). Then we have

$$Q_{wm.i,F}(\phi,\psi) = \frac{\sum[F(x) * i(x,y)]}{\sum F(x)}$$

where $*$ is multiplication. Recall that x is evaluation of ϕ, and y is evaluation of ψ. Summation is done over all the data objects, each object (i.e. experiment in the scope of our investigation) is taken exactly once. Accuracy of the computed mean can be outlined by a well known estimation of its variance

$$\begin{array}{ll} s_w^2 = & \frac{\sum F(x)}{(\sum F(x))^2 - (\sum F(x)^2)} * \sum[F(x) * (i(x,y) - Q_{wm,i,F})^2] \\ s_c^2 = & \frac{1}{(\sum F(x))-1} * \sum[F(x) * (i(x,y) - Q_{wm,i,F})^2] \end{array}$$

where s_w^2 is for $F(x)$ values taken as weights, s_c^2 is for $F(x)$ values interpreted as counts of occurrences. $Q_{wm,i,F}$ quantifiers are both weak and strong fuzzy multitudinal quantifiers on fuzzy data. Weak multitudinality is easy to see, $Q_{wm,i,F}$ used on crisp data is just $a/(a+b)$ for any $F(x)$ and $i(x,y)$. Strong multitudinality can be proved taking the conditions one by one. We explore several cases of these quantifiers. Continuous weighting functions which are zero valued at point $\{0, 0\}$, combined with Lukasiewicz and product implications lead to continuous quantifiers. It is not so in the case of Goedel implication. Let us take $F(x) = x$, the identity (label it $I(x)$). Then, we have

$$\begin{array}{ll} L \ldots \text{Lukasiewicz} & Q_{wm,L,I}(\phi,\psi) = \frac{\sum[x*\min\{1,1+y-x\}]}{\sum x} \\ P \ldots \text{Product} & Q_{wm,P,I}(\phi,\psi) = \frac{\sum \min\{x,y\}}{\sum x} \end{array}$$

where the summation is over all the data objects, as usually. Usage of $Q_{wm,P,I}$ leads to the fastest nontrivial computation if we consider $Q_{wm,i,F}$ quantifiers with F weightings which are 0 (1 respectively) valued in 0 (1 respectively) point only. Summations have to be done in sake of nontriviality and minimum is just one condition, i.e. faster than any numerical computation. Usage of Lukasiewicz or Goedel implications can be faster if the weighting function practically separates $[0, 1]$ interval onto two parts where objects are either fully neglected or used with maximum weight. Weighting means resemble IOWA operators [6], but they work in a different way. Results of their computations are equal to $Q_{wm,i,F}$ operators if weighting functions are identity in both models. There are problems with defining the IOWA operators if their weighting functions are not to be linear and there are data objects with formula evaluations $\{x_1, y_1\}$, $\{x_2, y_2\}$ where $x_1 = x_2$ and $y_1 \neq y_2$. In addition, there is violated stability of IOWA operators if they use nonlinear weighting functions.

It is sometimes possible to make fourfold table-like data which can be used for $a/(a+b)$ multitudinal quantifier computation. The value of a should be set to $\sum[F(x)*i(x,y)]$, value of b to $\sum[F(x)*(1-i(x,y))]$. It is something like the a value is conjunction evaluation summation of ϕ and of implication $\phi \to \psi$. The value of b is computed in a similar way, but the implication $\phi \to \psi$ is replaced by its negation. Another notable feature comes from $Q_{wm,P,I}$ term. It is $\sum T_{\text{Goedel}}(x,y)/\sum x$, i.e. it uses t-norm evaluation. We can use other t-norms and find their initial (pseudo)implication functions (say i-functions). Usage of product t-norm leads to constant initial i-function. It is the case B in the model squares overview (see Figure 2). Usage of Lukasiewicz t-norm leads to an i-function which falls to class of models represented by the case C, Figure 2. There is one difference, all the contour lines lead to $\{0, 1\}$ point of the evaluation square.

Important properties of quantifiers are changes of their evaluations with respect to formula alterations. Let us suppose that a chosen multitudinal quantifier has its output high enough on ϕ, ψ pair, say 0.9 or higher. Can it be on $-\phi, \psi$ or $\phi, -\psi$ too? Generally said, it can be so. Nevertheless, there are conditions which prevent the latter case (i.e. $\phi, -\psi$). Assume that the initial data separation (from $[-1, 1]$ interval into two $[0, 1]$ intervals) is done according to the default model at Figure 1 (i.e. case B, without interval overlap). If $Q_{wm,P,I}(\phi,\psi)$ is used and if its result is greater than 0.5, then the result of $Q_{wm,P,I}(\phi,-\psi)$ is lower than 0.5. It means that if ϕ multitudinates ψ by $Q_{wm,P,I}$, then ϕ does not multitudinate $-\psi$ (by the $Q_{wm,P,I}$). Somewhat weaker rule holds for $Q_{ws,L,F}$, it needs its F to be zero valued on some $[0, c]$ interval. Situation is less satisfactory when we use overlap models. It is necessary to use weighting functions with zero valuations on intervals greater than $[0, c_{overlap}]$. It practically turns such models into the model without overlapping.

Multitudinal quantifiers on crisp data which are frequently considered, are sample quantiles and statistical estimators of quantiles. Fuzzy sample quantiles on fuzzy data can be constructed by extending the classical ones. First, sort all the data objects according to their evaluations on a chosen model square, (i.e. $i(x, y)$, see Figures 2, 4). Second, assign a rod to each data objects. Set length of each rod to weight value of the object, (i.e. to $F(x)$). Concatenation of the rods results in a final staff. Requested quantile interval can be gained by multiplying the final length with the quantile and measuring out it from the end of the lowest $i(x, y)$ values. It does not depend on choice of particular sorting ranks of objects with equal $i(x, y)$ values. These quantifiers, label them $Q_{sq,i,F}$ (sq for sample quantile), reduce to classical sample quantiles if the used weighting function is replaced with constant function which has its output equal to 1. Quantifiers $Q_{sq,i,F}$ are strong fuzzy multitudinal quantifiers on fuzzy data. Weak multitudinality is evident, zero value weighted objects disappear from the final staff. Strong multitudinality is not hard to check too.

Statistical quantile estimators on crisp data have direct (and commonly known) counterparts on fuzzy data if output values of weighting functions are just 0 and 1. In such a case we utilize the fact that any continuous distribution function taken as random variable has uniform distribution on $[0, 1]$ interval. Quantile estimation reduces to testing on binomial distribution. It is given by

$$\sum_{i=0}^{k-1} \binom{n}{i} * (1-p)^{n-i} * p^i \leq \alpha$$

where $1-\alpha$ is confidence interval for stating that $100*(1-p)$ percents of objects have their $i(x, y)$ values greater than $i(x, y)$ of the k-th least object (from totally n objects). Label these quantifiers with $Q_{qe,p,\alpha,F2}$ (qe for quantile estimator). It is known that such quantifiers applied on crisp data are multitudinal. Hence, these quantifiers applied on fuzzy data are multitudinal too, since they neglect objects with 0 weight. It should be noted that we use special kind (i.e. two valued) of weighting functions herein.

Extension of $Q_{qe,p,\alpha,F2}$ onto estimations with general F weighting functions directly uses the process of quantile estimation, i.e. before it is summed into classical terms with binomial coefficients. Assume we estimate p-quantile (for $100*(1-p)$ percents of objects with high $i(x, y)$ values) and we have n objects. First, give each data object a rod which has its length set to $F(x)$ weight of the object. After it, data objects are sorted (as usually) according to their $i(x, y)$ evaluation levels and the staff is made from object rods. Rank the sorted objects, the first (with the lowest $i(x, y)$ value) has rank of 1, the last has rank of n. Put probabilities on each of the objects: p probability that it is below the p-quantile, (i.e. outside the searched interval), and $q = (1-p)$ probability that it is above the p-quantile, (i.e. inside the searched interval). Make all the combinations on possibilities of data objects lieings and do next procedure for each such combination. First, concatenate rods of those objects which have fallen outside the searched interval. Second, multiply probabilities which were selected in the applying combination. Assign it to the gained short staff. Finally, lay the short staff along the original staff to its end with the lowest values, as shown at Figure 5.

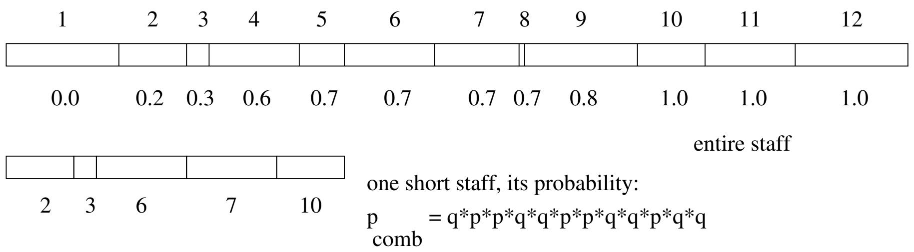


Figure 5: Staffs alignment during weighted quantile estimation.

After all the short staffs are laid, make subsequent summations. Take all short staffs which have their ends along particular rods into distinct groups. The intervals that belong to rods are open from their lower sides and closed from their upper sides. In such a manner, there is an initial part $[0, 0]$. It contains ends of short staffs which have zero lengths. Add up the assigned values which belong to short staffs from single groups. The outputs are group (rod) assigned values. Do a new succession along the entire staff from its end of the lowest values. Add up group assigned values until the summation strictly greater than α, the confidence value. Stop by the rod which is the first with the greater summation. Take its $i(x, y)$ value, label it i_k. The interval for p-quantile (of weighted occurrences) is $[i_k, 1]$ on $(1 - \alpha)$ confidence level. It means, the probability that p-quantile (of weighted occurrences) lies outside $[i_k, 1]$ is less than or equal to α. We assume that x values of the sample are representative enough with respect to x-dependent changes of $i(x_{fixed}, y)$ distributions. Weighting functions which have zero outputs on some $[0, c]$ interval, are a bypass if there is a problem with that assumption on $i(x, y)$ for small x values.

Label the newly gained quantifiers with $Q_{qe,p,\alpha,FG}$ (FG for general F weighting functions). Usage of two (i.e. $\{0, 1\}$) valued weighting functions reduces $Q_{qe,p,\alpha,FG}$ quantifiers into $Q_{qe,p,\alpha,F2}$ quantifiers. We can prove that $Q_{qe,p,\alpha,FG}$ are strong fuzzy multitudinal quantifiers on fuzzy data, by using the evaluation process that is exposed above. It directly implies, first, that $Q_{qe,p,\alpha,F2}$ are strong multitudinal quantifiers too, and second, that both $Q_{qe,p,\alpha,FG}$ and $Q_{qe,p,\alpha,F2}$ are weak multitudinal quantifiers as well. The presented quantifiers take $F(x)$ values as weights. It is possible to take $F(x)$ outputs as counts of occurrences. It leads into different quantifiers, label them $Q_{qI,p,v,FG}$ (v for border value, I explained in next). Such quantifiers are yielded by the usage of regularized beta function $I_p(r, s) = B(p, r, s)/B(r, s)$, where $B(r, s)$ is the beta function and $B(p, r, s)$ is the incomplete beta function. Incomplete Beta function is defined by integral: $B(p, r, s) \equiv \int_0^p t^{r-1}(1-t)^{s-1}dt$ and $B(r, s) = B(1, r, s)$. Cumulative probabilities for a binomial distribution can be computed by $I_p(r, s)$ with r, s integers. We just extend the definition onto arbitrary nonnegative real values, there is no other change. Set $f = \sum_1^n F(x)$ and $g = \sum_1^k F(x)$, where n is number of data objects and k is number of objects which have their $i(x, y)$ strictly lower than a chosen value v. Then we have

$$Q_{qI,p,v,FG} = 1 - I_{(1-p)}(f - g, g + 1)$$

where p is for the p-quantile, and $Q_{qI,p,\alpha,FG}$ expresses confidence interval for the p-quantile being in $[i_{k+1}(x, y), 1]$. It is straightforward to prove that $Q_{qI,p,v,FG}$ are strong fuzzy multitudinal quantifiers on fuzzy data.

Now, let us intrigue on residual implication of Goedel t-norm. That implication is somewhat special because it depends on x in just one condition checking. The situation resembles survival monitoring. There are developed methods for survival estimation. We exploit some of them to develop new multitudinal quantifiers. We use conception that x represents observation time (before censoring takes place) and y represents survival time for each data object. Each object is taken at time of $\min(x, y)$. No object is considered again after its censoring time (if it was censored). An object is taken as failed one if y is sharply less than x. An object is taken as censored if y is greater than or equal to x. It changes the notion of our version of survival analysis from "to live over" (to survive) into "to live up to" (to reach). The function, say $p(t)$, which is searched, is probability p of reaching an age t. We alternate classical Kaplan-Meier estimator to fit our model. Multiplier terms which are used for estimation of reaching (i.e. survival in the original case) probability, are gained by a method shown next. Consider computation at a time t_i when an event occurs. Count all objects which reached the time t_i (including those which fail at that time). It is the amount of objects with $\min(x, y)$ greater than or equal to that time t_i. Label this value with a_i. Count number of objects which failed at time t_{i-1} of previous event, label this value with b_i. The multiplier term is $a_i/(a_i + b_i)$, label it with r_i. Reaching probability at the time t_i is $\prod_{j \leq i} r_j$, set $p(t_i)$ to this value. The value of $p(t_i)$ is sample estimation of reaching for interval $(t_{i-1}, t_i]$. Estimation of $p(0)$ is set to 1, and of $p(t)$ is set to 0 for $t \in (t_n, 1]$ if t_n is the last time of an event occurrence and $t_n < 1$. Estimated mean of reaching time is gained by integrating the p function from 0 to 1. Label this estimator by $Q_{r1,C}$. We have considered all the objects with the same stress. It corresponds to setting the weighting function to constant of 1. Label generalized estimators which use weighting functions F, with $Q_{r1,F}$. Both $Q_{r1,C}$ and $Q_{r1,F}$ are classes of fuzzy quantifiers which are weak multitudinal. Neither of them is strong multitudinal quantifier. The cases of $r1$ quantifiers belong to model A at Figure 6.

It is possible to alter $Q_{r1,F}$ so that they can hold strong multitudinality. One solution is in two changes. First, b_i is count of all the objects which failed before time t_i. Second, $p(t_i)$ is computed directly as $a_i/(a_i + b_i)$. Then, it is used as value for interval $(t_{i-1}, t_i]$. In fact, $p(t)$ can be computed as $a/(a + b)$ for every time. End of the modified algorithm is the same as in the case of $Q_{r1,F}$. We have new quantifiers, label them

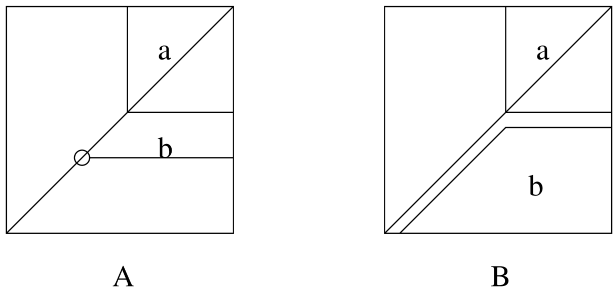


Figure 6: Model squares for reaching time quantifiers

with $Q_{r2,F}$ and their version with constant weighting function as $Q_{r2,C}$. The newly gained quantifiers $Q_{r2,F}$ and $Q_{r2,C}$ are strong fuzzy multitudinal quantifiers on fuzzy data. The cases of $r2$ quantifiers belong to model B at Figure 6. It is notable that $p(t)$ remains reach estimator and it has a natural property, it is nonincreasing function of time. Thus, we have developed both strong multitudinal quantifiers and weak multitudinal quantifiers which are not strong.

We have several classes and concrete examples of multitudinal quantifiers on fuzzy data. In addition, we need formulas to take advantage of the quantifiers. We have developed methods to generate formulas and their evaluations in previous section. Formulas which are used in data mining technics, are usually simple. We consider formulas which are generated by usage of conjunction, disjunction, the outer operation opposite, and constants 0 and 1. Multitudinal quantifiers on formulas ϕ, ψ can resemble implication $\phi \rightarrow \psi$. We look for formula pairs which lead into high results by the multitudinal quantifiers usage on them. In the same time, we do not want noninteresting formula pairs, as is in classical GUHA method. Thus we start from potentially the most interesting formulas. We try other weaker formula pairs until the multitudinal quantifiers are satisfied high enough on them. The starting pair$\{\phi, \psi\}$ is $\{1, 0\}$. We weaken the first position formulas into conjunctions of basic formulas and of their opposites. The second position formulas are weakened into disjunctions of basic formulas and of their opposites. Fitting formulas can be strengthen by subsequent disjunctions (conjunctions respectively) applied on the first (the second respectively) formulas of the formula pairs.

Let us restrict onto crisp data. It is known that a pair $\{$conjunction (ϕ, χ), $\psi\}$ which leads into high quantifier output, can be used for creation of a pair $\{\phi$, disjunction (negation(χ), ψ)$\}$ and the new pair leads into high quantifier output too. Let us come back to fuzzy data. We replace negation by opposite operation. Then, the relation between formula pairs which is shown above, does not hold in the field of mining on fuzzy data. It is caused by the fact that opposite of formula which has low evaluation value, can have low evaluation value too.

4. Conclusion

We have new methods to make data mining on fuzzy data. They are aimed to be used primarily for extraction of information from microarray experiments data. There are specific features of such data. The most considerable one is great number of genes, i.e. properties. It leads into too great amount of formulas. It can be solved by clustering the data and by usage of the resulting clusters in place of original genes.

Results of quantifier usage on the microarray data can be further used in several ways. They are among others: first, exploiting gene regulations, second, illnesses classification, third, gene expression prediction. Consider the first case. Formula pairs of form $\{\phi_1 \& \phi_2 \& \ldots \& \phi_k, \psi_1\}$, $\{\phi_1 \& \phi_2 \& \ldots \& \phi_k, \psi_2\}$, ..., $\{\phi_1 \& \phi_2 \& \ldots \& \phi_k, \psi_l\}$ can help in search for upstream regulatory sequences of ψ_1, ψ_2, ..., ψ_l. Consider the second case. Formula pairs which have conditional fitting, are to be searched if there are several classes of data objects. General formula pairs which fit in some classes but not in some others, are important cases. Object classification is frequently done by nearest neighbor search. Implication functions (together with weights) on objects can be used as other points in object vectors (i.e. cells in data rows).

Consider the third case. Formula pairs of form $\{\phi_{11}\&\phi_{12}\& \ldots \&\phi_{1k_1}, \psi\}$, $\{\phi_{21}\&\phi_{22}\& \ldots \&\phi_{2k_2}, \psi\}$, ..., $\{\phi_{l1}\&\phi_{l2}\& \ldots \&\phi_{lk_l}, \psi\}$ can be used for prediction of expression of genes in position of ψ formula.

We believe that GUHA inspired data mining technics are fruitful instruments in a broad range of practical applications. Life sciences and medicine produce high amount of experimental data and there are data collections which are available on internet. We will implement described methods and test them on gene expression data.

References

[1] B. De Baets, J. Fodor, "Residual operators of uninorms", *Soft Computing*, vol. 3, pp. 89-100, 1999

[2] T. R. Golub *et al.*, "Multiclass cancer diagnosis using tumor gene expression signatures", *PNAS*, vol. 98, pp. 15149-15154, 2001

[3] P. Hájek, T. Havránek, "Mechanizing hypothesis formation (mathematical foundations for a general theory)", Springer-Verlag Berlin-Heidelberg-New York, 1978

[4] M. Holeňa, "Fuzzy hypotheses for GUHA implications", *Fuzzy Sets and Systems*, vol. 98, pp. 101-125, 1998

[5] J. Quackenbush, "Microarray data normalization and transformation", *Nature Genetics*, vol. 32, pp. 496-501, 2002

[6] R. R. Yager, D. P. Filev, "Induced ordered weighted averaging operators", *IEEE Transactions on Systems, Man, and Cybernetics (Part B)*, vol. 29, pp. 141-150, 1999

Neural Network Approximation and Regularization Theory

doktorand:
TEREZIE ŠIDLOFOVÁ
Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, P.0. Box 5, 182 07 Prague 8, Czech Republic
terka@cs.cas.cz

školitel:
VĚRA KŮRKOVÁ
Institute of Computer Science, Academy of Sciences of the Czech Republic
Pod vodárenskou věží 2, P.0. Box 5, 182 07 Prague 8, Czech Republic
vera@cs.cas.cz

obor studia:
Teoretická Informatika

Abstract

We express the problem of approximating a data set $z = \{(x_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$ in the form of minimizing a functional that composes of an empirical error part and a regularization part. We prove that by adding the regularization part the problem becomes solvable. We derive existence and uniqueness of the solution. We also describe the shape of the minimizing function and show that it is in the form of a one-hidden layer feed-forward neural network with activation functions derived from the regularization part.

1. Introduction

Learning from data usually means to fit a function to a set of data $z = \{(x_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$. Typically it is not necessary that the function approximates the data exactly, we need some generalisation. There are infinitely many functions that fulfill this demand, so it is needed to precise the task. Additional constrains are put on the desired function such as continuity or smoothness, see for example [1], [3].

Mathematical expression of these ideas lies in formulating a functional, that would among admissible functions pick the one, that is reasonably close to the data and also corresponds to global property assumptions. The article deals with a special type of the functional based on Fourier transform and we show existence, uniqueness and even the form of the solution.

2. Preliminaries

A real or complex Banach space $(X, ||.||)$ is a vector space over real or complex numbers which is complete in the topology generated by the norm $||.||$, defined on X. A Hilbert space is a Banach space in which the norm is given by an inner product $\langle .,. \rangle$, that is $\|x\| = \langle x,x \rangle^{1/2}$. Sequences of elements of spaces are denoted by $\{x_n\}$ meaning $n \in \mathbb{N}_+$, where $\mathbb{N}_+$ is the set of positive integers.

The Banach space $X^*$of bounded (real-valued) linear functionals on X is called the dual or adjoint space. It defines weak convergence on X. A sequence $\{x_n\} \in X$ converges weakly to x ($x_n \rightharpoonup x$) if and only if $\lim_{n\to\infty} |f(x_n) - f(x)| = 0$ for each fixed $f \in X^*$.

$\mathbb{R}$ denotes the set of real numbers. For a positive integer d, $\Omega \subseteq \mathbb{R}^d$ we denote by $(C(\Omega), \|.\|_C)$ the space of continuous functions on Ω with the maximum norm and by $(\mathcal{L}_p(\Omega), \|.\|_p)$ the space of p-times integrable functions on Ω, where $\|f\|_p = \int_\Omega |f(x)|^p \mathrm{d}x$.

A functional $\mathcal{F} : X \to (-\infty, +\infty]$ is called *proper* if it is not identically equal to $+\infty$. The set $\operatorname{dom} \mathcal{F} = \{f \in X : \mathcal{F}(f) < +\infty\}$ is called the *domain* of $\mathcal{F}$.

$\mathcal{F}$ is *continuous* at $f \in \operatorname{dom} \mathcal{F}$ if for all $\epsilon > 0$ there exists $\delta > 0$ such that for every $g \in \operatorname{dom} \mathcal{F}$, $\|f - g\| < \delta$ implies $|\mathcal{F}(f) - \mathcal{F}(g)| < \epsilon$. The *modulus of continuity* of $\mathcal{F}$ at f is the function $\alpha_f : \mathbb{R}_+ \to \mathbb{R}_+$ defined as $\alpha_f(t) = \sup\{|\mathcal{F}(f) - \mathcal{F}(g)| : f, g \in \operatorname{dom} \mathcal{F}, \|f - g\| \le t\}$.

A functional $\mathcal{F}$ is *lower semicontinuous* in a topology if and only if $\{f; \mathcal{F}(f) > a\}$ is open for each real a. A functional is *sequentially lower semicontinuous* if and only if the convergence of $\{f_n\}$ to f implies $\mathcal{F}(f) \le \liminf_{n\to\infty} \mathcal{F}(f_n)$. Thus a functional $\mathcal{F}$ is *weakly sequentially lower semicontinuous* if and only if $f_n \rightharpoonup f$ implies $\mathcal{F}(f) \le \liminf_{n\to\infty} \mathcal{F}(f_n)$.

A functional $\mathcal{F}$ is *convex* on a convex set $C \subseteq \operatorname{dom} \mathcal{F}$ if for all $f, g \in E$ and all $\lambda \in [0,1]$, $\mathcal{F}(\lambda f + (1-\lambda)g) \le \lambda\mathcal{F}(f) + (1-\lambda)\mathcal{F}(g)$. $\mathcal{F}$ is *uniformly convex* on a convex set $C \subseteq \operatorname{dom} \mathcal{F}$ if there exists a non-negative function $\eta : \mathbb{R}_+ \to \mathbb{R}_+$, such that $\eta(0) = 0, \eta(t_0) > 0$ for some $t_0 > 0$ and for all $f, g \in E$ and all $\lambda \in [0,1]$, $\mathcal{F}(\lambda f + (1-\lambda)g) \le \lambda\mathcal{F}(f) + (1-\lambda)\mathcal{F}(g) - \lambda(1-\lambda)\eta(\|f-g\|)$. Any such function η is called a *modulus of convexity* of $\mathcal{F}$. The functional $\mathcal{F}$ is called *strictly uniformly convex* on C if $\eta(t) > 0$ for all $t \in \mathbb{R}_+$. Functional $\mathcal{F}$ is *(strongly) quasi-convex* if for all $f, g \in E$ it holds: $\mathcal{F}\left(\frac{1}{2}f + \frac{1}{2}g\right) \ (<) \le \max\{\mathcal{F}(f), \mathcal{F}(g)\}$.

Let X, Y be Banach spaces and $\mathcal{F} : X \to Y$ be a mapping from X to Y. We define the Gateaux derivative of $\mathcal{F}$ in f in direction h as $\mathrm{D}_f\mathcal{F}(h) = \lim_{t\to 0} \frac{\mathcal{F}(f+th)-\mathcal{F}(f)}{t}$. If the limit is uniform in h, we call the derivative the Frechet derivative (which will be our case). We can analogously define the second and so on derivatives.

3. Approximating a Data Set Formulated as a Minimization Problem

The task to find an optimal solution to the setting of approximating a data set $z = \{(x_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$ by a function from a general function space X is ill posed. A standard method to cope with ill-posed prolems is to impose additional (regularization) conditions on the solution. These are typically things like a-priori knowledge, or some smoothness constraints. The solution f_0 has to minimize a functional $\mathcal{F} : E \to \mathbb{R}$ that is composed of the error part and the "smoothness" part:

$$\mathcal{F}(f) = \mathcal{E}_z(f) + \gamma\Phi(f),$$

where $\mathcal{E}_z$ is the error functional depending on the data $z = \{(x_i, y_i)\}_{i=1}^N \subseteq \mathbb{R}^d \times \mathbb{R}$, Φ is the regularization part — the so called stabilizer and γ is the regularization parameter giving the trade-off between the two terms of the functional to be minimized.

An error functional is usually of the form:

$$\mathcal{E}_z(f) = \sum_{i=1}^{N} V(f(x_i), y_i)$$

A typical example of the empirical error functional is the classical mean square error:

$$\mathcal{E}_z(f) = \frac{1}{N}\sum_{i=1}^{N}(f(x_i) - y_i)^2.$$

In [3] a special stabilizer based on the Fourier Transform was proposed:

$$\Phi_G(f) = \int_{\mathbb{R}^d} \frac{|\hat{f}(s)|^2}{\hat{G}(s)} \mathrm{d}s,$$

where the hat indicates the Fourier Transform ($\hat{f}(s) = -(\frac{1}{2\pi})^{\frac{d}{2}} \int_{\mathbb{R}^d} f(x) e^{-ixs} \mathrm{d}x$, further we omit the constants) and $\hat{G}$ is some positive function tending to zero as $\|s\| \to \infty$. That means $1/\hat{G}$ is a high-pass filter.

Now we can define the functional $\mathcal{F}_G$ that is to be minimized:

$$\mathcal{F}_G(f) = \mathcal{E}_z(f) + \Phi_G(f) = \frac{1}{N} \sum_{i=1}^{N} (f(x_i) - y_i)^2 + \int_{\mathbb{R}^d} \frac{|\hat{f}(s)|^2}{\hat{G}(s)} \mathrm{d}s.$$

4. Existence and Uniqueness of the Solution

In this section we will derive conditions that will guarantee existence and uniqueness of solution to the above minimization problem.

We start with uniqueness. To solve this problem we need some convexity assumptions. We use the following result from [2, p. 15]:

Theorem 3 *A strongly quasi-convex functional $\mathcal{G}$ can achieve its minimum over a convex set C at no more than one point.*

Proof: If $\mathcal{G}(f_1) = \mathcal{G}(f_2) = \inf_{f \in C} f(x)$, then $1/2 f_1 + 1/2 f_2 \in C$, but

$$\mathcal{G}(1/2 f_1 + 1/2 f_2) < \max\{\mathcal{G}(f_1), \mathcal{G}(f_2)\} = \inf_{f \in C} \mathcal{G}(f),$$

which is a contradiction. □

We prove that under a natural assumption functional $\mathcal{F}_G$ is even strictly uniformly convex. This clearly implies strong quasi-convexity and using the previous theorem, we conclude, that the minimum of $\mathcal{F}_G$ (if it exists) is unique.

Theorem 4 *Suppose $\sup_{s \in \mathbb{R}^d} \hat{G}(s) \leq M < \infty$. With the notation from section 3, functional $\mathcal{E}_z$ is convex, the functional Φ_G is strictly uniformly convex with modulus $\eta(t) = t^2/M$ (in L_2). Hence, also $\mathcal{F}_G$ is strictly uniformly convex with modulus $\delta(t) = \frac{\gamma}{M} t^2$.*

Proof: For the first part, $\mathcal{E}_z(f)$ is sum of N elements, each of which is a convex functional, as (real) function $z \mapsto \frac{1}{N}(z - y_i)^2$ is convex.

To deal with the other functional, choose $\lambda \in [0, 1]$. We have to prove

$$\Phi_G(\lambda f + (1 - \lambda) g) \leq \lambda \Phi_G(f) + (1 - \lambda) \Phi_G(g) - \lambda(1 - \lambda) \eta(\|f - g\|)\,,$$

after rewriting

$$\lambda \Phi_G(f) + (1 - \lambda) \Phi_G(g) - \Phi_G(\lambda f + (1 - \lambda) g) \geq \lambda(1 - \lambda)(\|f - g\|)^2/M\,.$$

It is straightforward to check that the left-hand side of this inequality is equal to

$$\lambda(1 - \lambda) \Phi_G(f - g)\,.$$

Denote $h = f - g$. As $\hat{G}(s) \leq M$, we have

$$\Phi_G(h) = \int_{\mathbb{R}^d} \frac{|\hat{h}(s)|^2}{\hat{G}(s)} \mathrm{d}s \geq \frac{1}{M} \int_{\mathbb{R}^d} |\hat{h}(s)|^2 \mathrm{d}s = \frac{1}{M} \|\hat{h}\|_2^2 .$$

Since Fourier transform is an isometry on L_2, the last quantity is equal to $\frac{1}{M}\|h\|_2^2$, hence Φ_G is uniformly convex.

This means that $\gamma\Phi_G$ is uniformly convex with γ-times larger modulus, i.e. with modulus δ, and so is $\mathcal{F}_G = \mathcal{E}_z + \gamma\Phi_G$, as claimed. □

So we have proven uniqueness of the solution to the minimization problem.

To prove existence we use two basic results of approximation theory, see [2, p. 7-13]:

Theorem 5 *A weakly sequentially lower semicontinuous functional $\mathcal{F}$ defined on a weakly sequentially compact set C has an infimum f_0 such that $\mathcal{F}(f_0) = \inf_{f \in C} \mathcal{F}(f) = \min_{f \in C} \mathcal{F}(f)$.*

Weak lower sequential semicontinuity of a functional can be secured by several means, as for example by:

Theorem 6 *A convex functional $\mathcal{F}$ that has first and second derivatives at all points of an open convex set C (subset of E) is weakly sequentially lower semicontinuous in C.*

To apply Theorem 6 we have to prove the derivatives of $\mathcal{F}_G$ to exist.

Theorem 7 *Let C be an open convex subset of $\operatorname{dom} \mathcal{F}_G$. Functional $\mathcal{F}_G$ is weakly sequentially lower semicontinuous on C.*

Proof: Let us have a look at the regularization part. We compute the first derivative:

$$\begin{aligned}
\mathrm{D}_f \Phi_G(h) &= \lim_{t \to 0} \int_{\mathbb{R}^d} \frac{\left(\int_{\mathbb{R}^d} [f(x) + th(x)] e^{-ixs} \mathrm{d}x\right) \overline{\left(\int_{\mathbb{R}^d} f(\check{x}) + th(\check{x}) e^{-i\check{x}s} \mathrm{d}\check{x}\right)}}{t\hat{G}(s)} \\
&\quad - \frac{\left(\int_{\mathbb{R}^d} f(x) e^{-ixs} \mathrm{d}x\right) \overline{\left(\int_{\mathbb{R}^d} f(\check{x}) e^{-i\check{x}s} \mathrm{d}\check{x}\right)}}{t\hat{G}(s)} \mathrm{d}s \\
&= \int_{\mathbb{R}^d} \frac{\int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left(f(x)\overline{h(\check{x})} + h(x)\overline{f(\check{x})} \right) e^{-ixs} e^{i\check{x}s} \mathrm{d}x \mathrm{d}\check{x}}{\hat{G}(s)} \mathrm{d}s \\
&= \int_{\mathbb{R}^d} \frac{2\mathrm{Re}\left\langle \widehat{f}, \widehat{h} \right\rangle}{\hat{G}(s)} \mathrm{d}s
\end{aligned}$$

where $\mathrm{D}_f \Phi_G(h))$ means the first derivative of Φ_G in f in direction h.

Now we compute the second derivative:

$$\mathrm{DD}_f \Phi_G(h, k) = \lim_{t \to 0} \int_{\mathbb{R}^d} \frac{2\mathrm{Re}\left\langle \widehat{f + tk}, \widehat{h} \right\rangle}{t\hat{G}(s)} \mathrm{d}s - \frac{2\mathrm{Re}\left\langle \widehat{f}, \widehat{h} \right\rangle}{t\hat{G}(s)} \mathrm{d}s = \int_{\mathbb{R}^d} \frac{2\mathrm{Re}\left\langle \widehat{k}, \widehat{h} \right\rangle}{\hat{G}(s)}$$

where $\mathrm{DD}_f \Phi_G(h, k)$ is the second derivative of Φ_G in f in directions h, k.

Now we will need also the error part derivative (recall the error part is of the form ($\mathcal{E}_z(f) = \frac{1}{N}\sum_{i=1}^N (f(x_i) - y_i)^2$):

$$\mathrm{D}_f \mathcal{E}_z(h) = \frac{1}{N} \lim_{t \to 0} \frac{\sum_{i=1}^N (f(x_i) + th(x_i) - y_i)^2 - \sum_{i=1}^N (f(x_i) - y_i)^2}{t} =$$

$$= \frac{1}{N} \sum_{i=1}^N (2f(x_i)h(x_i) - 2h(x_i)y_i)$$

The second derivative is:

$$\mathrm{DD}_f \mathcal{E}_z(h,k) = \frac{1}{N} \lim_{t \to 0} \frac{\sum_{i=1}^N (2(f+tk)(x_i)h(x_i) - 2h(x_i)y_i) - \sum_{i=1}^N (2f(x_i)h(x_i) - 2h(x_i)y_i)}{t} =$$

$$= \frac{1}{N} \sum_{i=1}^N 2k(x_i)h(x_i)$$

By Theorem 6 $\mathcal{F}_G$ is weakly sequentially lower semicontinuous. □

Hence using theorems 5 and 7 we obtain existence of the solution to the minimization problem.

5. The Form of the Solution

We can describe the shape of the solution using a well known fact from mathematical analysis, see for example [3]:

Theorem 8 *Let the functional $\mathcal{F}$ defined on a set E in a Banach space X be minimized at a point $f_0 \in E$, with f_0 an interior point in the norm topology. If $\mathcal{F}$ has a derivative $\mathrm{D}\mathcal{F}_{f_0}$ at f_0, then $\mathrm{D}\mathcal{F}_{f_0} = 0$.*

The existence and uniqueness of the solution have been proven, so we can use Theorem 8 to derive the form of the solution. The proof of the following theorem is a rigorous version of the proof sketched in [3].

Theorem 9 *With the notation from section 3, functional $\mathcal{F}_G$ defined on real functions and for $\hat{G}$ symmetric we have a unique minimizing function f_0 of the form*

$$f_0(x) = \sum_{i=1}^N c_i G(x - x_i) + \sum_{k=1}^M d_k \psi_k(x),$$

where x_i are the data points and $\{\psi_k\}_{k=1}^M$ is the bias of the null space of functional Φ_G.

Proof: The existence and uniqueness of the solution has been proven in section 4. Now we will rewrite the functional $\mathcal{F}_G$ in terms of Fourier transform of f, $g = \hat{f}$:

$$\mathcal{F}_G(g) = \frac{1}{N} \sum_{i=1}^N (\int_{\mathbb{R}^d} g(s) e^{i x_i s} \mathrm{d}s - y_i)^2 + \gamma \int_{\mathbb{R}^d} \frac{|g(s)|^2}{\hat{G}(s)} \mathrm{d}s.$$

Let us observe, that $|g(s)|^2 = g(s)\overline{g(s)} = g(s)g(-s)$ because for f real it holds $\overline{\widehat{f(x)}} = \widehat{f(-x)}$. Now we will compute the derivative of $\mathcal{F}_G$ in g in direction h:

$$\mathrm{D}\mathcal{F}_{G_g}(h) = 2\frac{1}{N} \sum_{i=1}^N (\int_{\mathbb{R}^d} g(s) e^{i x_i s} \mathrm{d}s \int_{\mathbb{R}^d} h(s) e^{i x_i s} \mathrm{d}s - y_i \int_{\mathbb{R}^d} h(s) e^{i x_i s} \mathrm{d}s) +$$

$$+ \gamma \int_{\mathbb{R}^d} \frac{g(s)h(-s) + g(-s)h(s)}{\hat{G}(s)} \mathrm{d}s.$$

Now we put $\mathrm{D}\mathfrak{F}_{Gg_0}(h) = 0$ (g_0 is the minimizing function). This has to hold for all h. Let us take $h(s) = \delta_t(s)$, where δ_t means the delta-function in t. We obtain:

$$\mathrm{D}\mathfrak{F}_{Gg_0}(h) = 2\frac{1}{N}\sum_{i=1}^{N}(\int_{\mathbb{R}^d} g_0(s)e^{ix_is}\mathrm{d}s\, e^{ix_it} - y_ie^{ix_it}) + \gamma\frac{g_0(-t)}{\hat{G}(-t)} + \gamma\frac{g_0(-t)}{\hat{G}(t)} =$$

$$= 2\frac{1}{N}\sum_{i=1}^{N}(f_0(x_i) - y_i)e^{ix_it} + 2\gamma\frac{g_0(-t)}{\hat{G}(-t)} = 0,$$

as for $\hat{G}$ is symmetric. Now we multiply by e^{-ixt} and integrate over $\mathbb{R}^d$:

$$2\frac{1}{N}\sum_{i=1}^{N}(f_0(x_i) - y_i)\int_{\mathbb{R}^d} e^{i(x_i-x)t}\hat{G}(-t)\mathrm{d}t = -2\gamma\int_{\mathbb{R}^d} g_0(-t)e^{-ixt}\mathrm{d}t$$

Thus we obtain

$$\sum_{i=1}^{N}\frac{1}{N}(f_0(x_i) - y_i)G(x - x_i) = \gamma f_0(x)$$

and we see that the solution must be in the form

$$f_0(x) = \sum_{i=1}^{N} c_iG(x - x_i) + \sum_{k=1}^{M} d_k\psi_k(x),$$

where ψ_k are functions invisible to Φ_G. □

The solution derived is very nice, since it resembles a neural network with G as the activation functions shifted to the data points x_i. The problem of the number of hidden units being too large to be implemented can be solved by variable basis approximation using the obtained shape of the activation functions.

6. Conclusion

We have derived existence and uniqueness to the problem of finding a function close to the given data and simultaneously reasonably smooth (in terms of its Fourier transform). We showed that the solution is in the form of a one-hidden-layer feedforward neural network with activation functions depending on the form of regularization.

The drawback of this approach is that the obtained neural network has too many hidden units (as much as the number of data). This problem can be dealt by variable basis approximation limiting the number of hidden units. Nice approximation properties have been proven. This is unfortunately out of the scope of this article, so we kindly ask the reader to refer for example to works [5], [10].

The article was based on some ideas from [3].

References

[1] Cucker F., Smale S., On the Mathematical Foundations of Learning, *Bulletin of the American Mathematical Society* **39**, 1-49, 2001.

[2] Daniel J. W., *The Approximate Minimization of Functionals*, Prentice-Hall, Inc., 1971.

[3] Girossi F., Jones M., Poggio T., Regularization Theory and Neural Networks Architectures, *Neural Computation*, **7**, 219-269, 1995.

[4] Kůrková V., High-dimensional Approximation by Neural Networks, In *Advances in Learning Theory: Methods, Models and Applications*, Stuykens J. et. al. Ed., 69-88, Amsterdam, IOS Press, 2003.

[5] Kůrková V., Sanguinetti M., Error Estimates for Approximate Optimization by the Extended Ritz Method, *Research Report ICS-2002-882, Institute of Computer Science, Prague, 2002, submitted to SIAM Journal on Optimization*.

[6] Lukeš J., *Zápisky z funkcionální analýzy*, Karolinum, UK Praha, 2002.

[7] Lukeš J., Malý J., *Measure and Integral*, Matfyzpress, Praha, 1995.

[8] Rudin W. *Functional Analysis*, McGraw-Hill, USA, 1973.

[9] Wahba G., *Spline Models for Observational Data*, Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

[10] Zoppoli R., Sanguinetti M., Parisini T., Approximating Networks asn Extended Rutz Method for the Solution of Functional Optimization Problems, *J. of Optimization Theory and Applications*, **112**, 403-440, 2002.

Projekt LISp-Miner

doktorand:
ING. MILAN ŠIMŮNEK
KIT,VŠE Praha
simunek@cs.cas.cz

školitel:
DOC. RNDR. PAVEL DRBAL, CSC.
KIT, VŠE Praha
drbal@vse.cz

obor studia:
Informatika

Abstrakt

V následujícím textu je představen akademický projekt zaměřený na vývoj modulárního systému LISp-Miner pro data-mining. Tento systém je otevřený a volně dostupný a je vhodný pro výuku a výzkum v oblasti dobývání znalostí z databází. Projektový tým se skládá z učitelů i studentů z více vysokých škol. LISp-Miner může být snadno rozšiřován pomocí dalších modulů zaměřených na specifický způsob zpracování analyzovaných dat. Při tvorbě nových modulů jsou k dispozici rozsáhlé prostředky vytvořené při implementaci existujících modulů. Je možné vytvářet i specializovaná webová rozhraní uzpůsobená potřebám konkrétní analýzy. Přizpůsobení se může týkat jak rozsahu funkčnosti, tak použité terminologie. V článku jsou popsány některé existující moduly a zároveň jsou zmíněny i současné směry rozvoje systému.

1. Úvod

Automatizované zpracování provozních dat se v současné době stalo nezbytností téměř ve všech oblastech podnikatelské činnosti i veřejné správy. Vznikají tak rozsáhlé a složitě strukturované databáze zaznamenávající někdy i desítky let dlouhou historii činnosti organizací jako jsou banky, pojišťovny, výrobní podniky, obchodní řetězce ale i nemocnice, školy atd. Ve většině případů lze z těchto databází získat podstatně více informací a znalostí než bylo původně předpokládáno.

Proces, jehož cílem je získání takovýchto nových znalostí se v angličtině nazývá *Knowledge discovery in databases,* často se používá i termín *data mining*. Podle [5] je data mining *analýza (často rozsáhlých) observačních dat s cílem nalézt netušené vztahy a sumarizovat data novými způsoby tak, že jsou srozumitelná a užitečná pro jejich majitele.* Pro tento proces budeme používat termín *dobývání znalostí z databází* (zkráceně DZD). Slovo "dobývání" vyjadřuje fakt, že získání nových znalostí vyžaduje vynaložení rozsáhlého a nelehkého úsilí.

Dobývání znalostí z databází je předmětem jak rozsáhlého výzkumu a vývoje, tak i mnoha obchodních aktivit. Je k dispozici řada softwarových produktů a metodik pro DZD. Charakteristické je, že komerční systémy jsou značně drahé a těžko dostupné pro výuku. Dále je nutné poznamenat, že výzkum a vývoj v oblasti DZD je stále spíše na počátku.

Cílem článku je představit Projekt LISp-Miner. Projekt je zaměřený nejen na tvorbu samotného systému pro data-mining, ale i na vytvoření metodik celého procesu DZD a v neposlední řadě i na vysvětlení základních principů data-minigu ve výuce.

Systém LISp-Miner je vyvíjen na Fakultě informatiky a statistiky VŠE od roku 1996. Hlavním cílem je přiblížit studentům celý proces DZD, umožnit jim práci na vlastních analýzách a poskytnout i detailnější pohled na algoritmy a techniky použité při implementaci takového systému. Zároveň systém slouží jako platforma pro další výzkum v oblasti DZD a to jak na teoretické rovině, tak i v praktické prostřednictvím nových modulů vyvíjených zejména v rámci diplomových a disertačních prací. Nezanedbatelným přínosem je i možnost využít systém LISp-Miner pro DZD v oblastech, ve kterých nejsou finančně dostupné komerční systémy (např. při zpracování medicínských dat). LISp-Miner je volně ke stažení na adrese http://lispminer.vse.cz.

Systém LISp-Miner je výsledkem práce celé skupiny autorů, kteří se podíleli na jeho vývoji. Při jeho tvorbě byly použity dlouhodobé zkušenosti v oblasti data-miningu. Teoretické podklady byly publikovány v článcích a knihách již od roku 1960 (viz např. [2], [6], [4], [7], [3], [1], [14]). Starší verze některých modulů byly nejprve implementované jako samostatné aplikace jejich původními autory. V systému LISp-Miner je podstatným způsobem vylepšena funkčnost dříve existujících procedur, jsou přidány nové procedury a celý systém tvoří konzistentní celek těžící z propojeních jednotlivých dílčích modulů.

Hlavní rysy systému jsou spolu s přehledem jeho hlavních částí popsány v kapitole 2. Další směry vývoje jsou naznačeny v kapitole 3.

2. Systém LISp-Miner

LISp-Miner je modulární systém. Přehled současných modulů je na obrázku 1.

Moduly rozdělujeme do třech základních skupin:

- Seznámení s daty a jejich transformace
- Zpracování analýz
- Interpretace výsledků

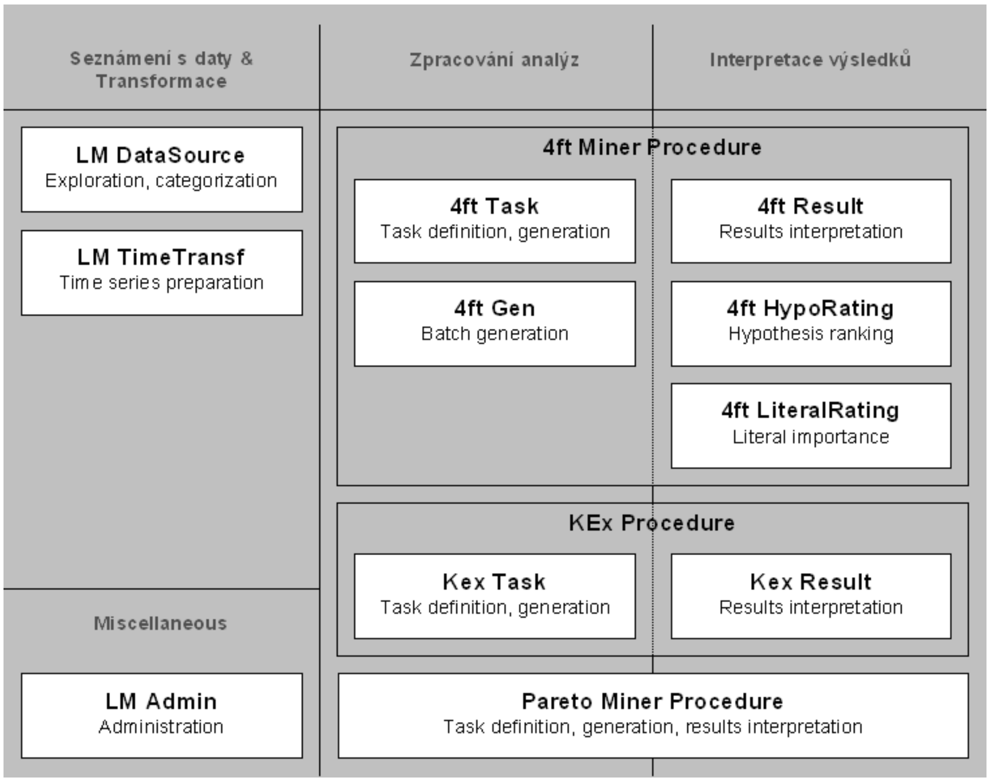
Toto rozdělení umožňuje udržet složitost každého modulu na úrovni přijatelné pro uživatele a zároveň pomáhá dodržet metodikami doporučovaný postup DZD (příkladem je metodika CRISP-DM).

Pro rychlé seznámení s daty slouží modul **LM DataSource**, mezi jehož funkce patří frekvenční analýza a široká škála transformací dat, podrobnosti jsou v odstavci 2.1. Do skupiny seznámení s daty a jejich transformace patří i modul LM TimeTransf [13]. Ten se zaměřuje na výpočet různých charakteristik časových řad tak, aby vypočítané charakteristiky mohly být snadno využity v dalších částech systému LISp-Miner.

Jako příklad analytické procedury uvádíme proceduru **4ft-Miner**, která je v hrubých rysech popsaná v odstavci 2.2. Jedná se o novou, oproti dřívějším značně rozšířenou, implementaci GUHA procedury ASSOC [3]. Při implementaci procedury 4ft-Miner byly využity dřívější zkušenosti [6], [7]. Současná implementace obsahuje další četná rozšíření a je integrovanou součástí celého systému LISp-Miner. Kromě procedury 4ft-Miner je součástí systému i procedura KEX [1] pro strojové učení, jejíž hrubý popis je v odstavci 2.3.

Prostředky vyvinuté během tvorby současné verze systému mohou být využity i v budoucnu a významným způsobem usnadní implementaci dalších analytických procedur.

Důležitou součástí systému je metabáze. Jedná se o centralizované úložiště dat ze všech modulů systému LISp-Miner. V metabázi se uchovávají nezbytné informace o analyzovaných datech (např. seznam relačních



Obrázek 1: Přehled současných modulů systému LISp-Miner.

tabulek, názvy a datové typy sloupců), definice transformací dat prováděných moduly LM DataSource a LM TimeTransf, zadání úloh jednotlivých procedur a v neposlední řadě i samotné výsledky analýz. Každý modul načítá veškeré své vstupy z metabáze a ukládá do ní i veškeré své výstupy. Omezená skupina modulů pracuje i přímo s analyzovanými daty, viz obrázek 2.

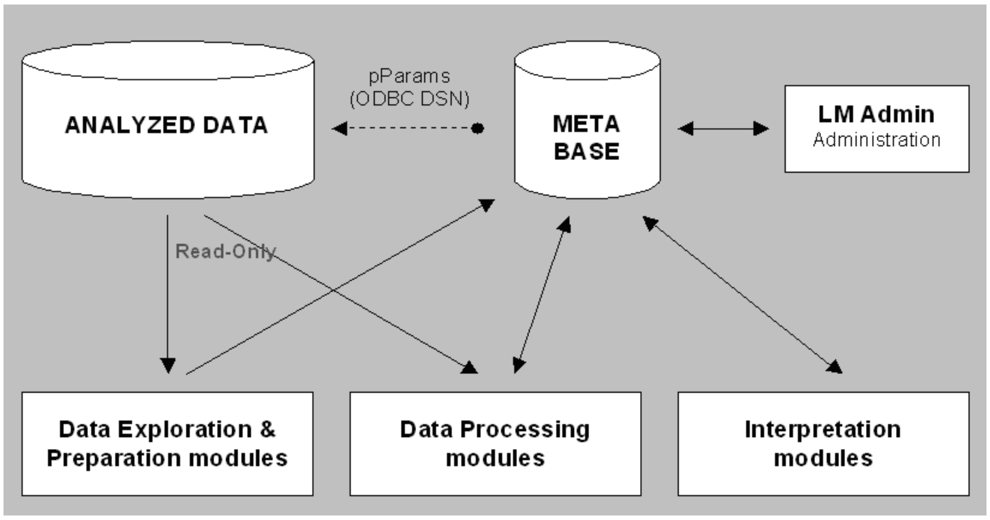
Uvedená architektura umožňuje snadno přidávat další moduly. Ty mohou využívat výstupy ostatních modulů jako své vstupy a své výstupy naopak zpřístupňovat jiným modulům pro další zpracování. Nový modul může být jak samostatnou aplikací, tak i webovým rozhraním, které spouští vybrané moduly v dávkovém režimu. Tak je možné velmi rychle vytvořit rozhraní specializovaná pro určitou oblast a konkrétní analýzu. V nich mohou být použity odborné termíny z analyzované oblasti a ovládání může být zjednodušené, aby s ním mohl snadno pracovat i samotný odborník na danou oblast. Takto lze rozšířit okruh lidí schopných provádět kvalifikované a hluboké analýzy dat, zejména pokud se jedná o opakované analýzy nad stále aktualizovanou databází. Rozhraní může být realizováno prostřednictvím WWW technologií. Jedna z takových implementací je zmíněna v [12].

2.1. LM DataSource

Modul LM DataSource slouží pro seznámení se s daty a zejména pro jejich přípravu pro další zpracování. V rámci seznámení je možné procházet seznam tabulek v analyzovaných datech, seznam sloupců v každé z nich a také data, která obsahují.

V rámci přípravy dat je možné definovat odvozené hodnoty z existujících sloupců tabulek. Pomocí databázových funkcí je možné dopočítat věk pacienta z data narození, celkovou výši platu jako součet základu a prémií nebo zjistit den v týdnu podle datumu uskutečnění telefonního hovoru. Přestože se jedná o odvozené informace, jejich použití může výrazně zlepšit výsledky analýzy.

Během přípravy dat je však klíčovou fází vhodná kategorizace hodnot. Může se jednat o vytvoření intervalů



Obrázek 2: Globální architektura systému LISp-Miner.

ze spojité veličiny, o seskupení okrajových hodnot do jedné kategorie, sloučení geograficky blízkých okresů do vyšších celků atd. Na obrázku 3 je ukázka dialogového okna, kterým se zahajuje automatická kategorizace sloupce Věk.

V současné době jsou k dispozici tyto volby pro automatickou tvorbu kategorií:

- **Enumerace** ... každá hodnota v analyzované relační tabulce je samostatnou kategorií,
- **Ekvidistantní intervaly** ... systém automaticky vytvoří intervaly o stejné délce s daným počátkem a s daným způsobem uzavření (zleva, zprava),
- **Ekvifrekvenční intervaly** ... systém automaticky vytvoří zadaný počet intervalů tak, že do každého intervalu spadá pokud možno stejný počet záznamů z analyzované tabulky.

V obrázku 3 je požadováno vytvoření intervalů o délce 10 uzavřených zprava. Ukázka výsledku je na obrázku 4.

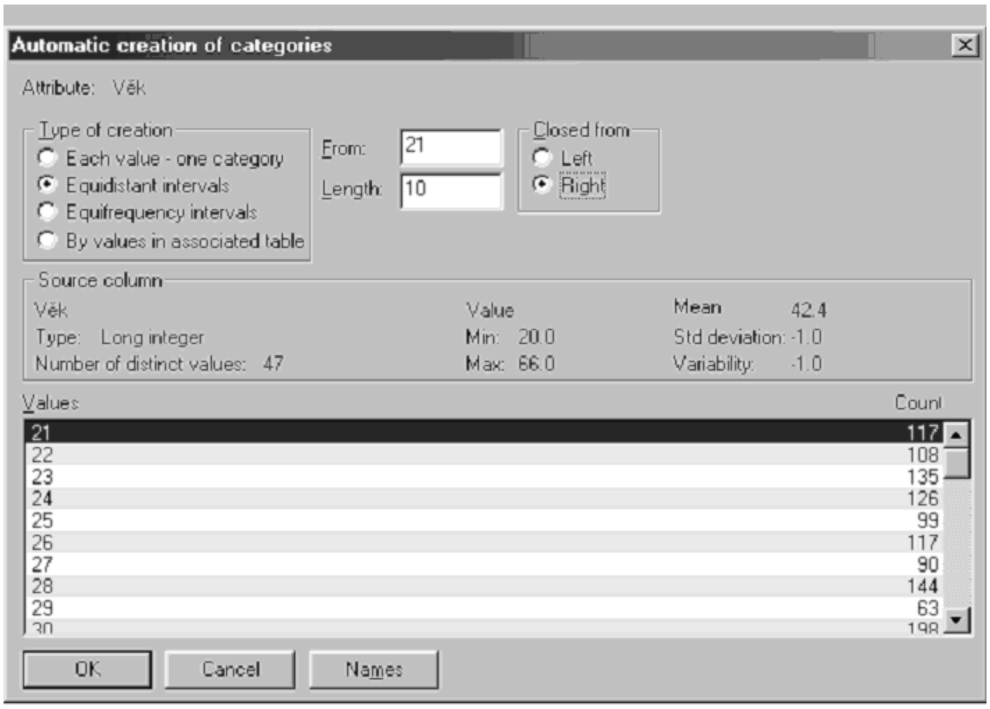
Důležitou funkcí modulu LM DataSource je i frekvenční analýza, která napomáhá vhodné kategorizaci hodnot. Ukázka frekvenční analýzy je na obrázku 5. Kategorie v seznamu i v grafu je možné řadit vzestupně/sestupně podle frekvence. Seznam je možné exportovat do schránky systému Windows a použít v dalších aplikacích. Analyzujeme-li najednou více atributů, jsou v seznamu i grafu uvedeny všechny možné kombinace kategorií ze všech analyzovaných atributů.

2.2. Procedura 4ft-Miner

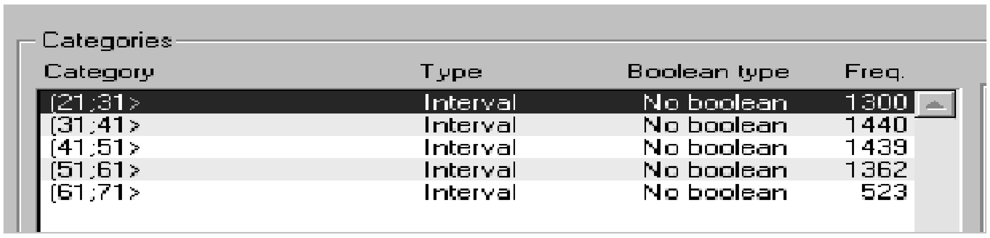
Procedura 4ft-Miner je nejstarší z procedur celého systému. Pro její implementaci existují široké teoretické podklady, které byly poprvé implementovány v souvislosti s metodou GUHA - [2], [6], [7]. Předchozí verze procedury 4ft-Miner je popsána např. v [10].

2.2.1 Asociační pravidla: Procedura 4ft-Miner nepoužívá klasický A-priori algoritmus. Namísto toho vyhledává platná asociační pravidla tvaru $\varphi \approx \psi$ nebo podmíněná asociační pravidla tvaru $\varphi \approx \psi/\chi$. Kde φ, ψ a χ jsou konjunkce booleovských atributů automaticky vytvořených z více-hodnotových atributů pomocí různých metod (viz dále).

Symbol $\approx$ se nazývá *4ft-kvantifikátor*. Asociační pravidlo $\varphi \approx \psi$ říká, že booleovské atributy φ a ψ jsou v



Obrázek 3: Ukázka okna pro kategorizaci.



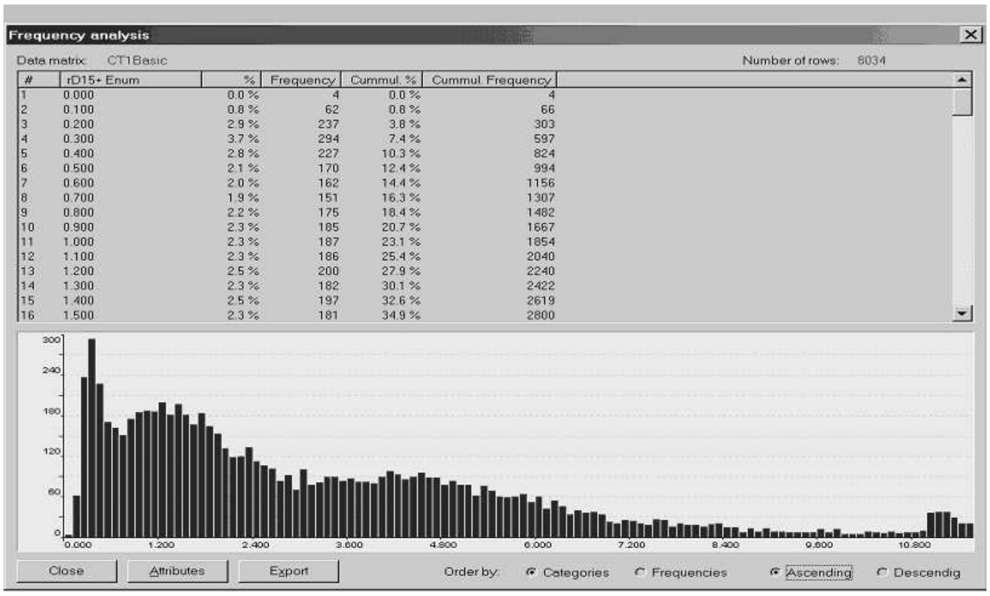
Obrázek 4: Ukázka intervalů vytvořených dle zadání z obrázku 3.

určité závislosti definované právě 4ft-kvantifikátorem $\approx$. Podmíněné asociační pravidlo $\varphi \approx \psi/\chi$ říká, že φ a ψ jsou v závislosti (ve smyslu $\approx$), jestliže je podmínka χ splněna.

Pro zadávání úloh je k dispozici jednoduché a uživatelsky přívětivé uživatelské rozhraní - viz obrázek 6.

2.2.2 Syntaktická bohatost: Hlavní rozdíl oproti A-priori algoritmu je schopnost 4ft-Mineru vytvářet a testovat syntakticky složitější asociační pravidla:

- Je možné používat podmíněná asociační pravidla ve tvaru $A \approx B/C$. To pomáhá hledat vztahy, kdy A je málo četná kombinace.
- Pravidla mohou obsahovat konjunkce atributů v libovolné své části - $A_1 \wedge A_2 \wedge ...A_n \approx B_1 \wedge B_2 \wedge ...B_n/C_1 \wedge C_2 \wedge ...C_n$.
- Každý atribut může obsahovat více než jednu hodnotu - např. A (Praha, Brno) znamená, že A může být buď "Praha" nebo "Brno".
- K dispozici je víc typů závislosti než pouhá implikace $A \Rightarrow B/C$. Celkem je v současné době k dispozici 17 *4ft-kvantifikátorů,* pomocí kterých lze vyjádřit hledaný vztah mezi A a B, viz např. [8].



Obrázek 5: Ukázka frekvenční analýzy pomocí modulu LM DataSource.

- Atributy v libovolné části asociačního pravidla mohou být seskupeny do celků a s těmi může být zacházeno najednou (nastavování parametrů, kopírování / přesouvání atd.).
- Během hledání platných asociačních pravidel jsou používány techniky pro redukci množství pravidel ve výsledku. Logicky odvoditelná pravidla mohou být automaticky vynechána, viz např. [8].

Detailnější rozbor procedury 4ft-Miner je uveden v [11].

Bohatost možností při zadávání úlohy vede k obrovskému množství pravidel, které je nutné vygenerovat a otestovat. Použité algoritmy jsou založeny na vhodné reprezentaci dat pomocí bitových řetězců a na velmi rychlých operacích nad těmito řetězci, viz např. [6], [11]. Výsledkem je velmi krátká doba nutná pro získání výsledků a používání procedury 4ft-Miner je interaktivní pro úlohy až do velikosti 10^7 generovaných asociačních pravidel.

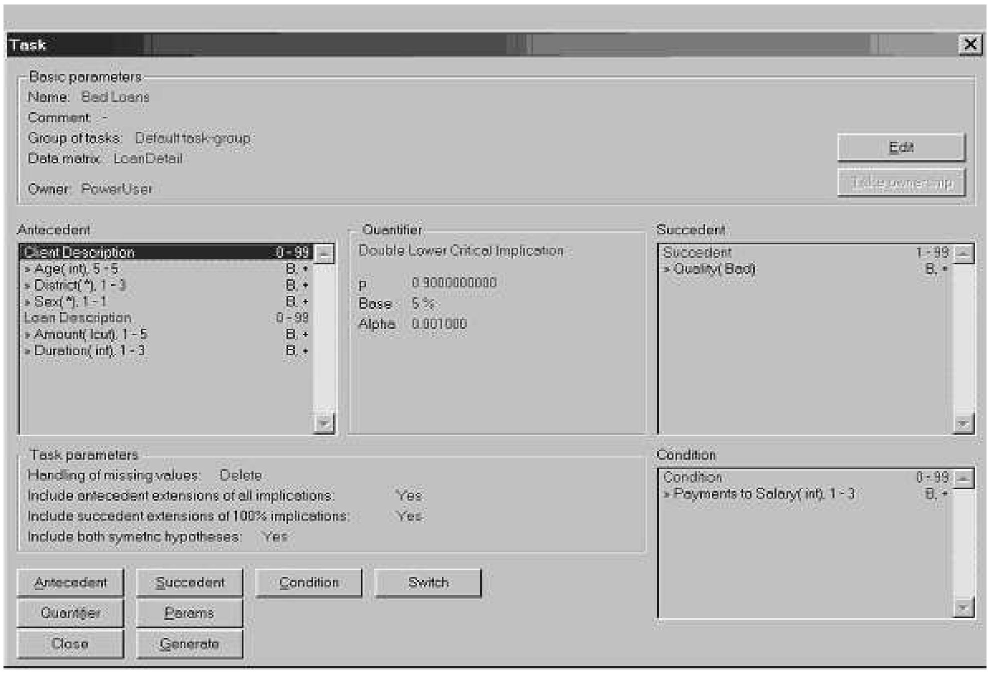
2.3. Expertní systém KEX

KEX je procedura strojového učení založená na asociačních pravidlech [1]. Verze implementovaná v systému LISp-Miner rozšiřuje schopnosti předchozích verzí zejména ve schopnosti sdílet vstupní i výstupní data s ostatními moduly zejména ve schopnosti používat výsledky fáze přípravy dat (z modulu LM DataSource).

2.3.1 Tvorba báze znalostí: Klíčovou funkcí expertního systému je tvorba báze znalostí. Cílem je vybrat takovou skupinu pravidel, která bude umožňovat co nejpřesnější klasifikaci nových případů. K tomu se používá trénovací sada případů, na které se může systém naučit. Oddělená testovací sada potom umožňuje otestovat kvalitu vytvořené báze znalostí.

Generovaná pravidla mají tvar $Ant \Rightarrow C$, kde:

- Ant je předpoklad (antecedent) ve tvaru $A(a) \wedge B(b) \wedge ... \wedge X(x)$
  - A, B, X jsou atributy
  - a, b, x jsou jejich kategorie (na rozdíl od procedury 4ft-Miner jsou povoleny pouze jednočlenné koeficienty)



Obrázek 6: Definice úlohy pro proceduru 4ft-Miner.

- C je závěr tvořený jedním atributem, podle kterého chceme případy klasifikovat

Při generování pravidel se postupuje od nejkratších k nejdelším (generování do šířky). Pro každé pravidlo se testuje, zda se jeho platnost významně odlišuje od závěru, kterých bychom získali složením vhodných pravidel již dříve vložených do báze znalostí. Pouze pravidla, která přinesou novou znalost, jsou vložena do báze.

Chceme například rozhodnout o vložení či nevložení pravidla

$$\text{Salary } \textit{(High)} \wedge \text{Age } \textit{(30ies)} \Rightarrow \text{District } \textit{(Praha)}$$

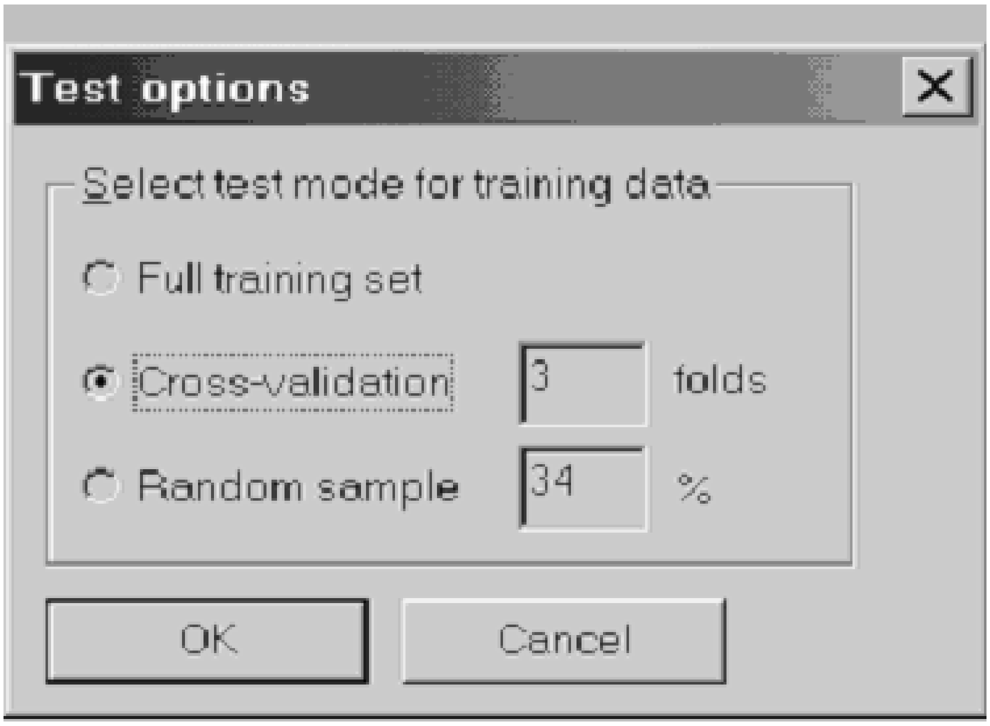
do báze znalostí. V tomto okamžiku jsme již (díky použitému algoritmu) rozhodli o těchto pravidlech:

- $\langle$*no antecedent*$\rangle \Rightarrow$ District *(Praha)*
- Salary *(High)* $\Rightarrow$ District *(Praha)*
- Age *(30ies)* $\Rightarrow$ District *(Praha)*

Z těchto pravidel vypočteme pomocí tzv. kombinační funkce (viz [1]) tzv. naskládanou váhu, kterou porovnáme s platností aktuálně zkoumaného pravidla. Jestliže se od sebe významně liší (ve smyslu χ^2 testu na zadané úrovni α), je nové pravidlo vloženo do báze znalostí.

V okamžiku, kdy otestujeme všechna možná pravidla, bude báze znalostí tvořena množinou pravidel, která je schopna klasifikovat nové případy s požadovanou přesností a neobsahuje přitom redundantní pravidla.

Důležitou součástí tvorby báze znalostí je otestování její kvality. Podle teorie bylo implementováno několik možných způsobů testování, z nichž si uživatel může vybrat pro danou situaci nejvhodnější:



Obrázek 7: Výběr metody testování báze znalostí.

- **oddělená testovací data**, která nejsou použita při trénování;
- **cross-validace** ... rozdělení dat na pevný počet částí a postupné vytváření báze za použití pouze (n - 1) částí a otestování na zbývající;
- **náhodné rozdělení** ... automatické rozdělení dat na trénovací a testovací;
- **testování v trénovacích datech** ... testování ne stejných datech jako probíhalo trénování (nejméně spolehlivá metoda testování).

Výsledkem testování je tabulka obsahující všechny provedené i neprovedené klasifikace spolu s absolutním i relativním počtem správných a chybných klasifikací.

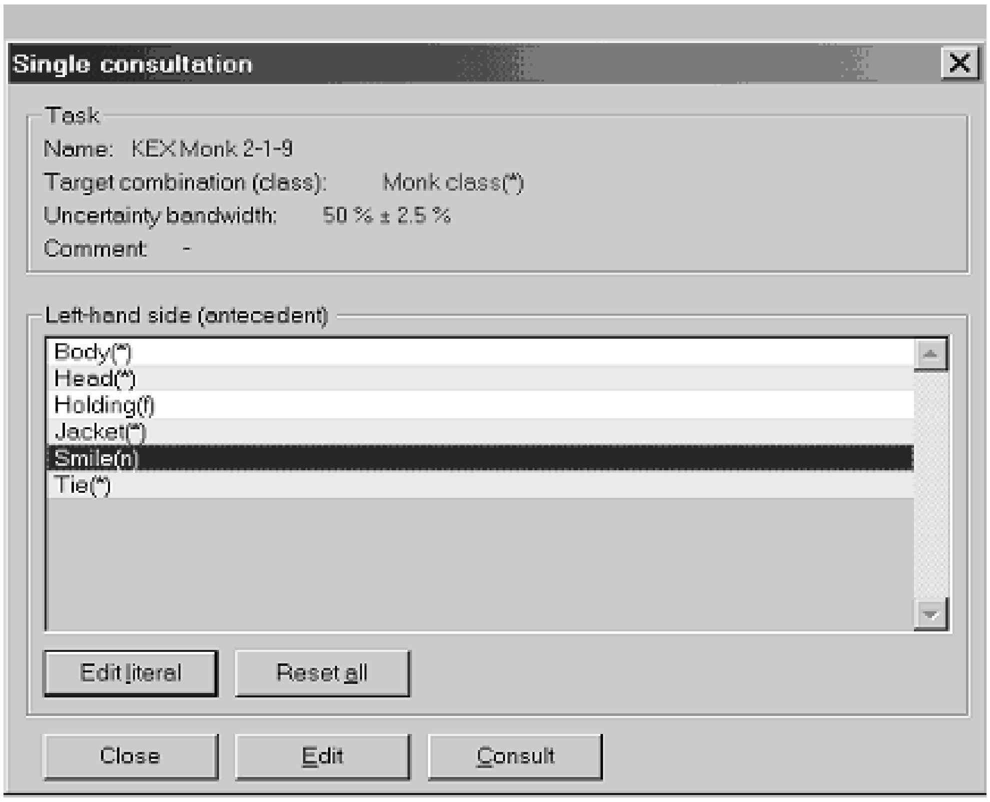
2.3.2 Konzultace: Po vytvoření báze znalostí je možné klasifikovat nové případy na základě známých hodnot atributů z antecedentu. Přitom není nutné znát hodnoty pro všechny atributy.

Každá klasifikace je doplněna mírou nejistoty se kterou byla udělána. Podle naskládané váhy z použitých pravidel jsou tyto možnosti: jistě, téměř jistě, pravděpodobně, možná. Jestliže známé hodnoty atributů nedovolují klasifikaci uskutečnit, může být odpovědí i odmítnutí klasifikace.

Při dávkové konzultaci určíme soubor, který obsahuje případy pro klasifikaci. Je-li součástí souboru i klasifikovaný znak, je na závěr konzultace uvedena tabulka shodná s tabulkou používanou pro testování báze znalostí. Tímto způsobem se je možné testovat bázi znalostí na vlastních testovacích datech.

3. Další vývoj

V současné době je připravována nebo již probíhá implementace několika dalších procedur. Ty budou využívat výstupů existujících modulů a při jejich implementaci budou použity algoritmy, datové struktury a knihovní funkce vyvinuté v rámci systému LISp-Miner.



Obrázek 8: Individuální konzultace.

Díky použité modulární architektuře je přidání nového modulu relativně snadné a je popsáno v jednotlivých částech projektové dokumentace. Novému modulu je umožněno sdílet data s ostatními moduly prostřednictvím metabáze.

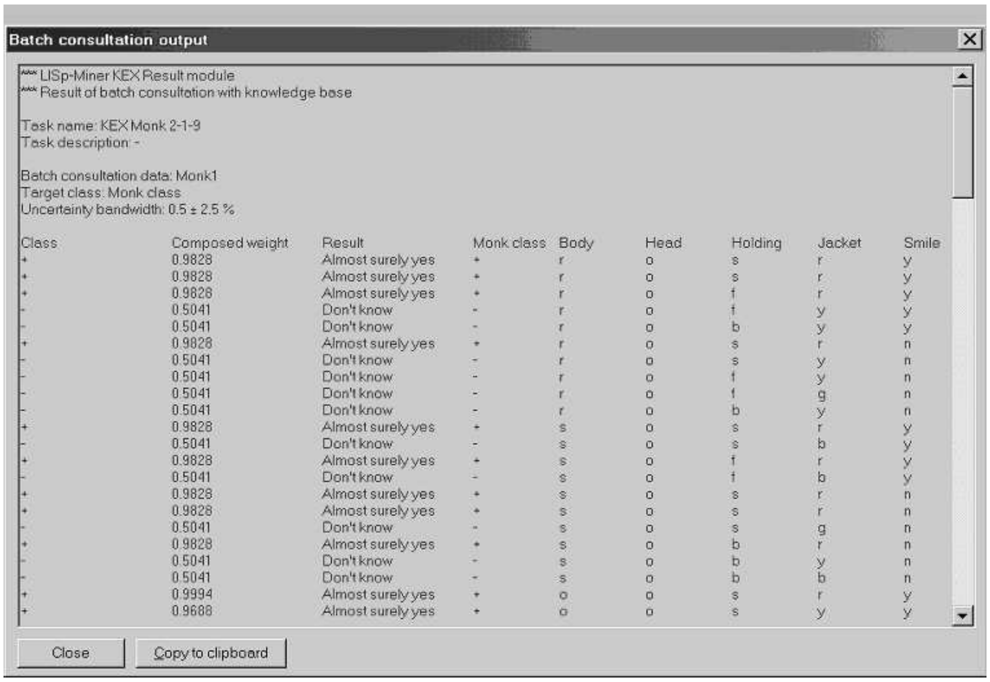
3.1. Nové procedury

Jednou z procedur s největším potenciálem je multi-relační data-mining. Ten je založen na předpokladu, že velké množství potenciálně zajímavých informací není skryto pouze v jedné databázové tabulce, ale ve vazbě mezi více tabulkami. Ty nejzajímavější informace nezískáme pouze z tabulky uskutečněných nákupů jednotlivých druhů zboží. Ani je nezískáme pouze z tabulky popisující detailní jednotlivé druhy zboží. Musíme právě tyto dvě tabulky nejprve spojit pomocí existujícího vztahu *1:n*. Potom získáme z tabulky prodejů mnoho agregovaných charakteristik (např. průměrný počet položek zakoupený pro jednotlivé druhy zboží, minimální měsíční tržby za jednotlivé obchody atd.). Tyto charakteristiky budou tvořit dodatečné atributy v tabulce popisující druhy zboží a umožní nám získat nové znalosti. K získání těchto charakteristik budou použity dříve implementované techniky bitových řetězců, které zaručí velmi rychlý výpočet.

Dalším směrem vývoje je hledání zajímavých závislostí mezi dvěma více-hodnotovými atributy. To vede k nutnosti rychlé práce s kontingenčními tabulkami o rozměru $K \times L$. Použití bitových řetězců vede opět k velmi efektivními způsobu výpočtu frekvencí v těchto složitých kontingenčních tabulkách.

3.2. Vizualizace výsledků

Výsledky libovolné data-miningové procedury musí být jednoduše interpretovatelné, aby použití této procedury mělo praktický smysl. Z praxe se ukazuje, že je vždy obtížné přeložit výsledky z formy používané pro počítačové zpracování zpět do pojmů a obratů oblasti, ze které pochází data. Z těchto důvodů je vhodné se zaměřit na vývoj co nejintuitivnějších způsobů zobrazení výsledků, kterým by rozuměl i počítačový laik. Místo seznamu hypotéz nebo kontingenčních tabulek a různých kvantifikátorů, může být pro uplatnění



Batch consultation output

*** LISp-Miner KEX Result module
*** Result of batch consultation with knowledge base

Task name: KEX Monk 2-1-9
Task description: -

Batch consultation data: Monk1
Target class: Monk class
Uncertainty bandwidth: 0.5 ± 2.5 %

Class	Composed weight	Result	Monk class	Body	Head	Holding	Jacket	Smile
+	0.9828	Almost surely yes	+	r	o	s	r	y
+	0.9828	Almost surely yes	+	r	o	s	r	y
+	0.9828	Almost surely yes	+	r	o	f	r	y
-	0.5041	Don't know	-	r	o	f	y	y
-	0.5041	Don't know	-	r	o	b	y	y
+	0.9828	Almost surely yes	+	r	o	s	r	n
-	0.5041	Don't know	-	r	o	s	y	n
-	0.5041	Don't know	-	r	o	f	y	n
-	0.5041	Don't know	-	r	o	f	g	n
-	0.5041	Don't know	-	r	o	b	y	n
+	0.9828	Almost surely yes	+	s	o	s	r	y
-	0.5041	Don't know	-	s	o	s	b	y
+	0.9828	Almost surely yes	+	s	o	f	r	y
-	0.5041	Don't know	-	s	o	f	b	y
+	0.9828	Almost surely yes	+	s	o	s	r	n
+	0.9828	Almost surely yes	+	s	o	s	r	n
-	0.5041	Don't know	-	s	o	s	g	n
+	0.9828	Almost surely yes	+	s	o	b	r	n
-	0.5041	Don't know	-	s	o	b	y	n
-	0.5041	Don't know	-	s	o	b	b	n
+	0.9994	Almost surely yes	+	o	o	s	r	y
+	0.9688	Almost surely yes	+	o	o	s	y	y

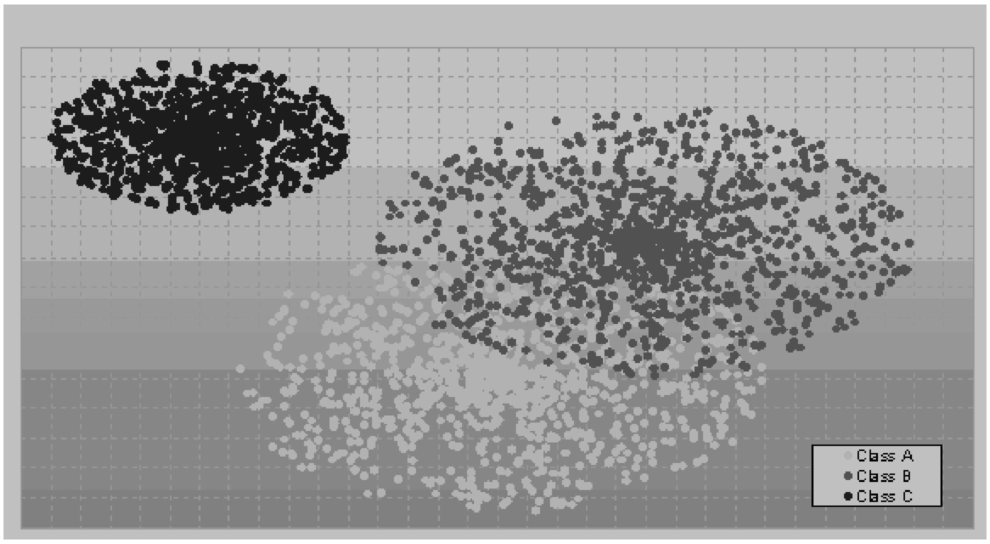
Close Copy to clipboard

Obrázek 9: Dávková konzultace.

výsledků více inspirující zobrazovat výsledky ve formě dvou- nebo dokonce tří-rozměrného grafu.

Na obrázku 10 je ukázán jeden z možných způsobů takového zobrazení. Je znázorněna skupina objektů rozdělených do třech tříd - A, B a C a zobrazených v dvourozměrném grafu podle dvou atributů - Salary na ose X a Age na ose Y. Zatímco třída C je jasně samostatná, třídy A a B se překrývají. To znamená, že atributy Salary a Age jsou dostatečné pro rozlišení objektů třídy C od objektů tříd A a B, zatímco nejsou dostatečné pro rozlišení mezi třídami A a B navzájem.

Naším cílem je vyvinout takové způsoby zobrazování výsledků, který budou snadno intuitivně interpretovatelné pro uživatele a zejména přímo pro odborníky ve zkoumané oblasti.



Obrázek 10: Vizualizace výsledků.

References

[1] Berka, P., Ivánek, J.: *Automated knowledge acquisition for PROSPECTOR-like expert systems,* In (Bergadano, de Raedt eds.) Proc. ECML'94, Springer 1994, pp. 339-342.

[2] Hájek, P., Havránek, T.: *Mechanising Hypothesis Formation - Mathematical Foundations for a General Theory,* Springer-Verlag, 1978, pp. 396.

[3] Hájek, P., Havránek, T., Chytil, M.: *Metoda GUHA,* Praha, Academia, 1983, pp. 314.

[4] Havránek, T.: *The present state of the GUHA software,* International Journal of Man-Machine Studies, 15, 1981, pp. 253-264.

[5] Hand, D., Manilla, H., Smyth, P.: *Principles of Data Mining,* MIT 2001.

[6] Rauch, J.: *Some Remarks on Computer Realisations of GUHA Procedures.* International Journal of Man-Machine Studies, 10, 1978, pp. 23-28.

[7] Rauch J.: *Main Problems and Further Possibilities of the Computer Realizations of GUHA Procedures,* International Journal of Man-Machine Studies, 15, 1981, pp. 283-287.

[8] [8] Rauch, J.: *Interesting Association Rules and Multi-relational Association Rules,* In Communications of Institute of Information and Computing Machinery, Taiwan. Vol. 5., No. 2, May 2002, pp. 77-82.

[9] Rauch, J., Šimůnek, M.: *Mining for 4ft Association Rules,* In Discovery Science 2000. Red. Arikawa, S. - Morishita S. Springer Verlag 2000, pp. 268-272.

[10] Rauch, J., Šimůnek, M.: *Mining for 4ft Association Rules by 4ft-Miner,* In INAP 2001, The Proceeding of the International Conference On Applications of Prolog. Prolog Association of Japan, Tokyo October 2001, pp. 285-294.

[11] Rauch, J., Šimůnek, M.: *Alternative Approach to Mining Association Rules,* In FDM02, The IEEE ICDM02 Workshop Proceedings (eds. T.Y. Lin, S. Ohsuga), IEEE, Maebashi December 2002, pp. 157-162.

[12] Černý, Z., Dolejší, P., Rauch, J., Šebek, M.: *Dobývání znalostí v medicínských datech - případová studie,* Přijato k prezentaci na konferenci Znalosti 2003.

[13] Šlesinger, J: *Předzpracování časových dat pro systém Lisp-Miner,* Přijato k prezentaci na konferenci Znalosti 2003.

[14] Zembowicz, R., Zytkow, J.: *From Contingency Tables to Various Forms of Knowledge in Databases,* Advances in Knowledge Discovery and Data Mining (eds. Fayyad, U. M. et al.), AAAI Press/ The MIT Press, 1996, pp. 329-349.

Elektronický zdravotní záznam a telemedicína

doktorand:
MGR. JOSEF ŠPIDLEN
Ústav Informatiky AV ČR, EuroMISE Centrum, Pod Vodárenskou věží 2, 182 07 Praha 8
spidlen@euromise.cz

školitel:
RNDR. ANTONÍN ŘÍHA, CSC.
Ústav Informatiky AV ČR, EuroMISE Centrum, Pod Vodárenskou věží 2, 182 07 Praha 8
riha@euromise.cz

obor studia:
Biomedicínská informatika

Abstrakt

Dizertační práce zkoumá možnosti modelování a reprezentace medicínské informace a tzv. elektronického zdravotního záznamu. Vzhledem ke klíčovému požadavku univerzálnosti a dynamické modifikovatelnosti množiny sbíraných typů je pro uchování informací navržena a matematicky popsána grafová struktura zahrnující tzv. znalostní bázi a datové složky. Kromě způsobu ukládání informací text popisuje třívrstvou architekturu systému a způsob získávání uložené informace pomocí definovaného aplikačního rozhraní. Na závěr jsou zmíněny možnosti mobilního přístupu k záznamu, využití v telemedicínských aplikacích a závěry plynoucí z prvních testování.

1. Úvod

1.1. Cíle dizertačního projektu

Elektronický zdravotní záznam má potenciál uzavřít cyklus mezi klinickou praxí, výzkumem a výukou. Vzhledem k náročnosti jeho obsahu i použití musí být založen na nejlepších výsledcích informatiky i počítačových technologií. Na druhé straně, protože nelze snadno opustit ani přepracovat dosavadní systémy natož čekat s tvorbou nových na dobu, kdy technologie i informatika pokročí natolik, aby bylo možno splnit všechny požadavky na takový záznam kladené, otevřenost a modulárnost použitých systémů umožňující integraci nehomogenních komponent a jednoduchou modifikovatelnost množiny uchovávaných dat zde nabývá nezastupitelného významu.

V projektu "Elektronický zdravotní záznam a telemedicína" se snažíme o zvládnutí současného stavu problematiky elektronického zdravotního záznamu i souvisejících oblastí a prostředků, které poskytují současné softwarové – zejména databázové – technologie a výsledky medicínské informatiky, a o jejich rozpracování ve směru dokonalejšího naplnění kladených požadavků.

Vzhledem k aplikované povaze tohoto výzkumu je cíl projektu směřován k vytvoření a otestování funkční distribuované prototypové aplikace. Těžiště práce je v prozkoumání vhodnosti využití různých technik uložení dat, otestování soudobých XML komunikačních trendů, zvolení vhodných metod pro začlenění multimediálních atributů do elektronického zdravotního záznamu a zvážení možností různého způsobu integrace formalizovaných lékařských doporučení. Celá práce je vytvářena s důrazem kladeným na podporu v telemedicínské oblasti, převážně je počítáno s možnostmi vzdáleného přístupu ke zdravotnímu záznamu pomocí Internetu a různých mobilních zařízení. Prototypovou platformou pro vývoj je Pocket PC 2002 a testovacím zařízením T-Mobile Mobile Digital Assistant.

1.2. Inspirace

Jedním z výzkumných směrů Evropského centra pro medicínskou informatiku, statistiku a epidemiologii – Kardio (EuroMISE Centra – Kardio) je aplikovaný interdisciplinární výzkum v různých oblastech medicínské informatiky. Součástí tohoto výzkumu jsou otázky reprezentace medicínských znalostí. Prototypová aplikace, na které pracujeme, je inspirována některými evropskými projekty z oblasti elektronického zdravotního záznamu (EHR), zejména projektem I4C/TripleC, na kterém EuroMISE centrum spolupracovalo a ve kterém vznikla dvouvrstvá pilotní aplikace EHR ORCA (Open Record for Care) [1]. Vyvíjená aplikace rozvíjí elektronický zdravotní záznam, který jsem implementoval v rámci své diplomové práce a je inspirována konzultacemi s lékaři – převážně kardiology – a českými, evropskými a mezinárodními standardy jako HL7, DASTA a komunikačními standardy ISO TC 215 a CEN TC 251 [2].

2. Architektura vyvíjeného EHR

2.1. Architektura MUDR

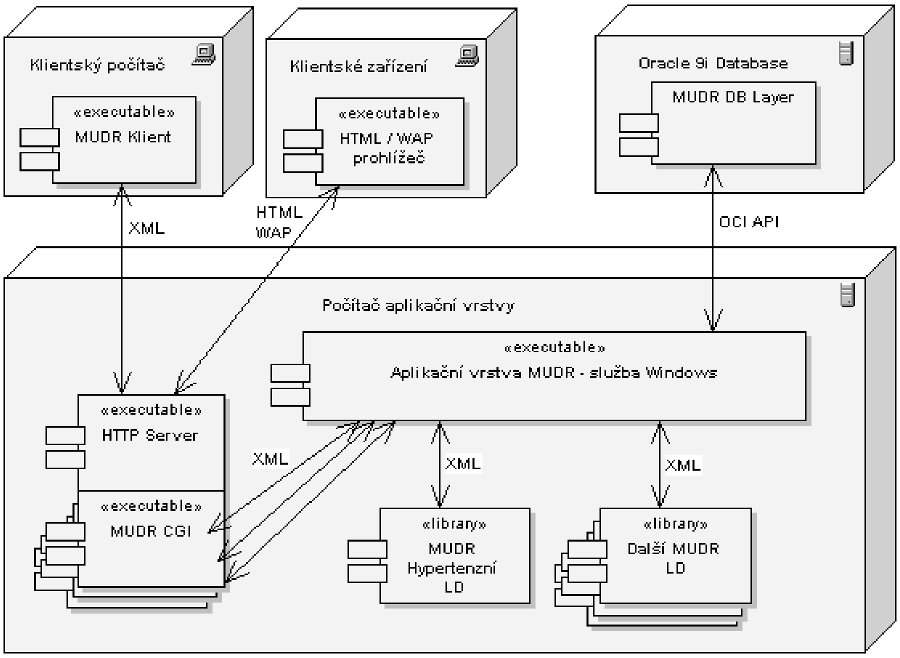
V rámci diplomové práce jsem pracoval na vývoji EHR MUDR (MUltimedia Distributed Record) [3], který je nyní modifikován a dále rozvíjen. Elektronický zdravotní záznam MUDR je založen na třívrstvé architektuře s datovou, aplikační a klientskou vrstvou. Tato dekompozice umožňuje oddělit jednotlivé funkční části systému a umožňuje uživatelským rozhraním pracovat na různých platformách. Funkce datové vrstvy spočívá v ukládání informací a kontrole základní referenční integrity dat. Aplikační (funkční) vrstva zajišťuje zjednodušený pohled na databázi a zpřístupňuje data uživatelským rozhraním a různým klientským aplikacím. Tito klienti komunikují s aplikační vrstvou pomocí MUDR API definovaného ve formě platných XML dokumentů odpovídajících příslušnému XML schématu. K přenosu tohoto XML je využito HTTP serveru na aplikační vrstvě. XML je na straně klienta zabaleno do HTTP POST žádosti a tím je přeneseno na vstup Gateway Interface (CGI) skriptu, který jej dále předává aplikační vrstvě.

Ve formě dynamických knihoven jsou ke službě aplikační vrstvy připojována lékařská doporučení. Jejich komunikace probíhá na úrovni win32 prostřednictvím exportovaných funkcí a sdílené paměti. Těmto funkcím je předáváno XML odpovídající MUDR API. V tomto ohledu získávají knihovny informace ze zdravotního záznamu pacienta stejným způsobem jako MUDR klienti. Architektura je zobrazena na obrázku číslo 1.

2.2. Architektura MUDR^2

Architektura EHR MUDR^2 vychází ze základní koncepce třívrstvé architektury MUDR, kterou dále dekomponuje způsobem zobrazeným na obrázku číslo 2. Datová vrstva MUDR^2 již nemusí být nutně implementována na databázovém stroji Oracle. Komunikace obecného MUDR DB Serveru s aplikační vrstvou je zajištěna pomocí adaptačních modulů – "MUDR DB Connection Module". Služba aplikační vrstvy – "MUDR Application Layer Service" – zvolí příslušný modul ve formě dynamicky připojitelné knihovny dle konkrétního databázového stroje. S využitím tohoto modulu již služba aplikační vrstvy komunikuje s datovou vrstvou plně transparentně.

Dále služba aplikační vrstvy MUDR^2 plní logicky obdobné funkce jako služba aplikační vrstvy MUDR; zajišťuje funkční logiku aplikace, připojuje knihovny lékařských doporučení (LD) apod. Rozdíl patrný na první pohled je ve zpřístupnění svých služeb. Pro komunikaci směrem ke klientům integruje služba aplikační vrstvy tzv. komunikační moduly. V první verzi probíhá implementace komunikačního modulu MUDR WS. Tento modul obsahuje a zpřístupňuje objekty implementující rozhraní "MUDR .NET Remoting API" (MUDRNRAPI). Pomocí RPC technologie založené na .NET Remotingu [4] je možno vzdáleně volat metody těchto objektů. Toto využívá další komponenta aplikační vrstvy – MUDR Web Service.



Obrázek 1: Schéma distribuované architektury EHR MUDR.

Tato webová služba zpřístupňuje klientům aplikační rozhraní "MUDR Web Service Application Interface" (MUDRWSAPI). Typický lékař tedy pracuje s k EHR MUDR ze své pracovní stanice, využívá aplikačního HTTP serveru pro komunikaci a SOAP protokolu pro kódování příkazů a parametrů a přistupuje takto ke službě MUDR Web Service [5].

Pro případné využití EHR MUDR tenkými klienty ve formě HTML či WAP klientů na jednoduchých zařízeních je dále počítáno s vytvořením tzv. MUDR WS Proxy Service. Tato komponenta ve formě CGI by měla vystupovat na jedné straně jako klient MUDR Web Service a na straně druhé by měla zpřístupnit EHR MUDR ve formě HTML či WAP stránek.

3. Reprezentace dat

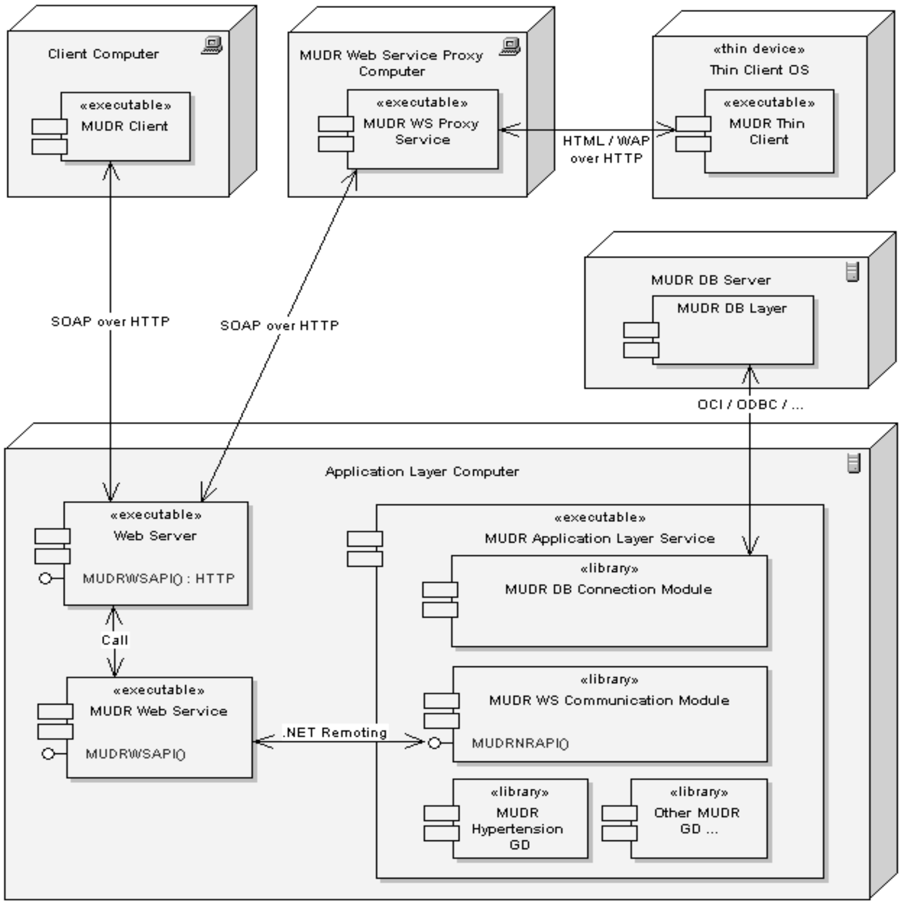
Z důvodů nezávislosti záznamu na použitém databázovém stroji nepopisuje práce fyzické schéma databáze. Místo toho je poskytnuto matematizované zobecnění, které je možno fyzicky implementovat pomocí různých databázových strojů různým způsobem. Některé databázové systémy umožňují využít hnízděných tabulek či objektově relačních technologií, v jiných systémech je třeba si vystačit s relačními tabulkami. Příslušnou transparentnost přístupu zajišťuje MUDR DB Connection Module.

Vzhledem k požadavku na dynamicky se měnící množinu sbíraných druhů údajů není možné využít jako základ pro zaznamenání hodnot klasickou relační tabulku se sloupečky odpovídajícími jednotlivým ukládaným veličinám. Místo toho jsou k ukládání dat použity dále popsané struktury teorie grafů, tzv. znalostní báze v kombinaci s datovými složkami.

3.1. Znalostní báze

Hlavním úkolem znalostní báze je zachycovat druhy sbíraných údajů a vztahy mezi nimi. Formálně znalostní bázi $MUDR\ KB$ definujeme jako orientovaný graf:

$$MUDR\ KB = (\mathbf{V}_{kb}, \mathbf{E}_{kb}). \tag{1}$$



Obrázek 2: Schéma architektury MUDR[2].

Vrcholy grafu, prvky $n \in \mathbf{V}_{kb}$, označujeme jako uzly znalostní báze. Každý uzel znalostní báze je čtveřice:

$$n = (\gamma, \varphi, \omega, \epsilon_{nd}), \tag{2}$$

kde γ je jednoznačný identifikátor v rámci všech uzlů, φ je mnemotechnický, maximálně desetiznakový, řetězcový identifikátor, ω je datový typ a ϵ_{nd} obsahuje základní administrativní údaje například o tom, kdo a kdy vložil uzel do znalostní báze.[1] Hrany grafu $e \in \mathbf{E}_{kb}$ jsou také čtveřice:

$$e = (\alpha, \beta, \tau, \epsilon_{ed}), \tag{3}$$

kde $\alpha, \beta \in \mathbf{V}_{kb}$ označují odkud kam hrana vede, τ je typem hrany určujícím druh vztahu mezi koncovými uzly a ϵ_{ed} uchovává administrativní údaje, tentokrát týkající se příslušné hrany.

Mezi typy hran existuje jeden dominantní typ – tzv. *inferior* – vedoucí od rodiče k potomkovi a určující tímto hierarchický vztah mezi uzly znalostní báze. Vyjmeme-li ze znalostní báze hrany všech ostatních typů, požadujeme, aby vznikl orientovaný les s několika málo stromy. Tyto stromy označujeme jako *domény znalostní báze*. Každá doména sdružuje uzly sloužící ke stejnému účelu. Typicky právě jedna doména slouží pro uchování množiny všech sbíraných údajů o pacientovi. Uzly této domény nazýváme *sémantické typy*.

[1] Dále si z oblasti programování vypůjčíme indexové značení, například zápisem $n[\omega]$ budeme myslet třetí složku, tj. datový typ uzlu n.

Množinu sémantických typů značíme $\mathbf{V}_s$; přirozeně platí $\mathbf{V}_s \subseteq \mathbf{V}_{kb}$. Jiné domény uchovávají například hierarchicky strukturovanou mezinárodní klasifikaci nemocí MKN10, anatomicko-terapeuticko-chemickou klasifikaci chemických názvů ATC či klasifikaci měrných jednotek SI nebo seznam léků dostupných na českém trhu. Hrany dalších typů umožňují vložit medicínskou znalost do této struktury. Představit si můžeme typy hran, které označují ekvivalenci dvou sémantických typů nebo například indikace a kontraindikace některého léku.

Uzly se stejným otcem nazýváme *bratry* a značíme $n_1 \diamondsuit n_2$. Formálně je-li $n_1, n_2 \in \mathbf{V}_{kb}$, pak:

$$n_1 \diamondsuit n_2 \overset{def}{\iff} \exists\, e_1, e_2 \in \mathbf{E}_{kb}, e_1[\alpha] = e_2[\alpha], e_1[\beta] = n_1, e_2[\beta] = n_2, e_1[\tau] = e_2[\tau] = \text{"}inferior\text{"}. \quad (4)$$

Triviálně vidíme, že bratrství je relací ekvivalence. Logicky, bratři slouží k podobným účelům, např. zaznamenání ulice a města v kontaktní adrese pacienta.

Uzly nemající otce nazýváme *kořeny domén* a pro $n \in \mathbf{V}_{kb}$ píšeme:

$$n \in \mathbf{R}_{kb} \overset{def}{\iff} (\forall e \in \mathbf{E}_{kb}, e[\tau] = \text{"}inferior\text{"} \implies e[\beta] \neq n). \quad (5)$$

Požadujeme, aby jméno φ bylo mezi bratry vždy jednoznačné. Stejnou jednoznačnost vyžadujeme mezi kořeny domén, tedy:

$$(n_1 \diamondsuit n_2) \,\vee\, (n_1, n_2 \in \mathbf{R}_{kb}) \implies (n_1 = n_2) \,\vee\, (n_1[\varphi] \neq n_2[\varphi]). \quad (6)$$

Datový typ uzlu je významný především pro sémantické typy. Rozlišujeme základní datové typy jako číslo, boolean či text, multimediální datové typy: obraz, audio, video a binární soubor a speciální, tzv. referenční datové typy. Dále existuje pomocný datový typ adresář. Datové složky (viz def. 8 na str. 137) mající datový typ adresář mají prázdnou hodnotu a slouží například k seskupování svých podsložek. V nejnovější verzi MUDR[2] je na multimediální data uplatňován jednotný pohled a rozlišování je prováděno podle "Multipurpose Internet Mail Extensions Type" (mime-type, viz RFC 2048) atributu. V této verzi byl také přidán výčtový datový typ "enum".

3.2. Datové složky

Konkrétní zadaná data tvoří orientovaný les *MUDR DF*, formálně:

$$\mathit{MUDR\ DF} = (\mathbf{D}_{df}, \mathbf{E}_{df}). \quad (7)$$

Záznamy jednoho pacienta odpovídají vždy jednomu stromu v tomto lese. Vrcholy $d \in \mathbf{D}_{df}$ nazýváme datové složky a definujeme jako čtveřice:

$$d = (\delta, \sigma, \lambda, \epsilon_{df}). \quad (8)$$

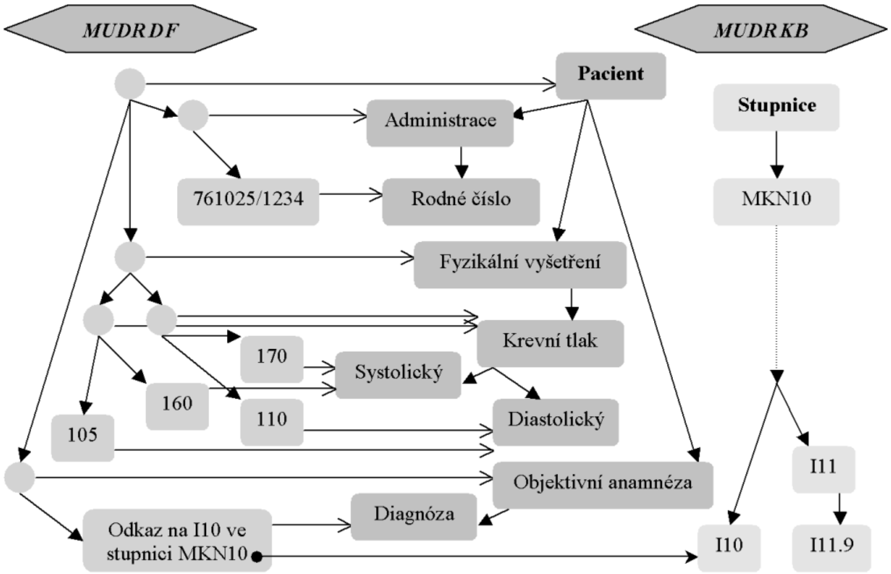
Zde δ je jednoznačný identifikátor v rámci všech datových složek, σ je sémantický typ ve znalostní bázi $(\sigma \in \mathbf{V}_s)^2$ a λ je tzv. hodnotou datové složky. Doména této hodnoty je určena datovým typem datové složky, který implicitně vyplývá z sémantického typu σ. Pod položkou ϵ_{df} se skrývá množství administrativních informací určujících kdo a kdy, zadal, smazal či potvrdil příslušnou datovou složku, jaké je období platnosti hodnoty, spolehlivost zadaného údaje apod.

[2] Implementace samotná je typicky prováděna formou odkazu na sémantický typ ve znalostní bázi.

Hrany $e \in \mathbf{E}_{df}$ jsou tentokrát netypované, vytvářejí pouze hierarchický vztah "otec - syn", tedy $\mathbf{E}_{df} \subseteq \mathbf{D}_{df} \times \mathbf{D}_{df}$. Navíc pro *MUDR DF* musí platit:

1. $\forall d \in \mathbf{D}_{df}\ (\exists d' \in \mathbf{D}_{df}, (d', d) \in \mathbf{E}_{df} \vee d[\sigma] \in \mathbf{R}_{kb})$
2. $d, d' \in \mathbf{D}_{df}, (d', d) \in \mathbf{E}_{df} \implies \exists e \in \mathbf{E}_{kb}\ e[\alpha] = d[\sigma], e[\beta] = d'[\sigma], e[\tau] = "inferior"$

Slovně popsáno první podmínka vyjadřuje, že každá datová složka má otce nebo její sémantický typ je kořenem domény. Druhá podmínka tvrdí, má-li datová složka d otce d', pak sémantický typ d je syn sémantického typu d'. Zjednodušeně řečeno, podmínky zaručují, že datová část odpovídá znalostní bázi a že v ní "neplavou" volné uzly. Malý příklad několika uložených dat a části znalostní báze je znázorněn na obrázku číslo 3.



Obrázek 3: Příklad reprezentace dat v EHR MUDR^2.

4. Získávání informací z MUDR

4.1. Rozhraní pro přístup k datům

Pro účely komunikace s aplikační vrstvou EHR MUDR bylo definováno aplikační rozhraní MUDR API. Toto rozhraní má formu platných XML dokumentů odpovídajících XML Schématu MUDRAPI.xsd [6]. Existují 2 druhy platných XML dokumentů: směrem k aplikační vrstvě je používán dokument sestávající z identifikace uživatele a posloupnosti příkazů, směrem od aplikační vrstvy je používán dokument složený z posloupnosti odpovědí. Každému příkazu je přiřazen jednoznačný identifikátor v rámci XML dokumentu. Tento identifikátor kopíruje aplikační vrstva do odpovědí, aby klient správně a jednoduše rozpoznal, k jakému příkazu se příslušná odpověď' váže. Ke každému příkazu jsou definovány možné odpovědi.

Aplikační rozhraní MUDR API bylo navrženo tak, aby obsahovalo relativně malou množinu příkazů a přitom umožňovalo plnohodnotně pracovat s elektronickým záznamem včetně úprav znalostní báze, správy uživatelů a jejich pravomocí, práce s výrazy v různých světových jazycích apod.

4.2. Typické použití aplikačního rozhraní

Jak je patrno z obrázku číslo 2 aplikační rozhraní pro EHR MUDR^2 je řešeno odlišně po technické stránce. Komponenta MUDR Web Service zpřístupňuje tzv. MUDRWSAPI. Toto aplikační rozhraní sestává z množiny funkcí, které je možno volat vzdáleně pomocí technologie webových služeb. Pro příkazy původního MUDR API nalezneme v MUDRWSAPI funkce provádějící obdobnou službu v kontextu nového EHR. Opačně toto pravda není. V původním EHR bylo analyzováno chování klientů a na základě získaných poznatků bylo API rozšířeno. Klienti nyní mohou požádat například o celý strom $T \subseteq MUDR\ DF$ odpovídající určitému pacientovi nebo získat určitý fragment znalostní báze $K \subseteq MUDR\ KB$. V původní implementaci získávali klienti postupně (maximálně "po patrech") jednotlivé datové složky $d \in \mathbf{D}_{df}$ či uzly znalostní báze $n \in \mathbf{V}_{kb}$ s přidanou informací o některých hranách v příslušné struktuře. Klienti implementují různé modifikace a kombinace klasických "Depth First Search" (DFS) a "Breadth First Search" algoritmů k prohledávání získaných grafových struktur. Celkem existuje 21 základních příkazů, z nichž pro potřeby prohledání informací ze záznamu pacienta jsou relevantní 4 příkazy uvedené v tabulce číslo 1.

Příkaz	**Popis příkazu**
get_knowledge_domains	Vypíše kořenové uzly znalostních domén.
get_knowledge_node	Vypíše detailní informace o uzlu znalostní báze.
get_node_neighbours	Vypíše seznam okolních uzlů k uzlu znalostní báze.
get_data_files	Vrátí určitou(-é) datovou(-é) složku(-y).

Tabulka 1: Vybrané příkazy aplikačního rozhraní MUDR API.

Na příkladě v jazyce C++ si ukážeme algoritmus načtení podstromu znalostní báze. Každé volání metody `AppendChildren` načte další patro stromu znalostní báze. Vnější cyklus přes `pN2` prochází uzly z patra, které právě zkoumáme. Vnitřní cyklus přes `pN1` přidává postupně do `xmlNextLayer` požadavky na prozkoumání všech synů. Je-li alespoň jeden takovýto požadavek přidán, je před koncem metody voláno zpracování `xmlNextLayer.ProcessCmnds()` a metoda `AppendChildren` je rekurzivně volána na výsledek. Objekty třídy `CMUDRXMLDoc` sdružují dva XML dokumenty, jeden shromažďuje příkazy a do druhého (`m_pXMLResp`) je uložena odpověď po `ProcessCmnds()`.

```
bool CKnowledgeTree::LoadTree(int nRootNodeID)
{
      CMUDRXMLDoc xmlDoc;
      xmlDoc.AppendCmnd_GetKnowledgeNode(nRootNodeID);
      if(xmlDoc.ProcessCmnds()) return AppendChildren(xmlDoc);
      else return false;
}

bool CKnowledgeTree::AppendChildren(CMUDRXMLDoc& xmlDocNodes)
{
      IXMLDOMElementPtr pELRoot = xmlDocNodes.m_pXMLResp->GetdocumentElement();
      IXMLDOMNodePtr pN1, pN2 = pELRoot->GetfirstChild();
      CMUDRXMLDoc xmlNextLayer;

      while(pN2)
      {
            pN1 = pN2->GetfirstChild(); pN2 = pN2->GetnextSibling();
            if(!pN1) continue;

            do {
                  CKnowledgeNode *pNode = new CKnowledgeNode(pN1);
                  StoreKnowledgeNode(pNode);

                  for(int i = 0; i <= pNode->m_dwaInfs.GetUpperBound(); i++) {
                        xmlNextLayer.AppendCmnd_GetKnowledgeNode(pNode->m_dwaInfs.GetAt(i));
                  }

            } while ((pN1 = pN1->GetnextSibling()));
      }

      if(!xmlNextLayer.IsEmpty()) {
            if(xmlNextLayer.ProcessCmnds()) return AppendChildren(xmlNextLayer);
```

```
            else return false;
        }

        return true;
}
```

Příkaz *get_data_files* lze použít třemi základními způsoby:

1. Dotaz na datovou složku $d \in \mathbf{D}_{df}$ pomocí jejího identifikátoru $d[\delta]$. Vrácena je kompletní informace o datové složce. V případě multimediálních dat lze dokonce specifikovat některé vlastnosti hodnoty datové složky $d[\lambda]$ a systém provede příslušné transformace jako například převod do určitého formátu či nastavení úrovně komprese.

2. Dotaz na datové složky pomocí jejich společného otce. Vráceny jsou všechny datové složky $d \in \mathbf{D}_{df}$, které mají otce $d' \in \mathbf{D}_{df}$ specifikovaného dotazem, tj. pro vrácené datové složky d platí $(d', d) \in \mathbf{E}_{df}$.

3. Vyhledávání datových složek $d \in \mathbf{D}_{df}$ pomocí zadané klíčové hodnoty λ', sémantického typu $d[\sigma]$ a upřesňujících parametrů vyjadřujících vztah mezi $d[\lambda]$ a λ'. Tento způsob je hodně flexibilní a lze jím vyhledat například datové složky systolického krevního tlaku většího než 180 mmHg nebo příjmení pacientů začínající řetězcem "Nov".

Při prohledávání datových složek i znalostní báze vyvstává potřeba odkazování se na určitý sémantický typ či libovolný jiný uzel znalostní báze. Jednou možností, jak se odkázat na uzel $n \in \mathbf{V}_{kb}$ je využít identifikátor $n[\gamma]$. Pro případ, že klient tímto údajem nedisponuje, lze využít tzv. *úplné jméno uzlu*. Úplné jméno uzlu n definujeme jako posloupnost $\varphi_0, \varphi_1, ..., \varphi_k$, jestliže existují uzly $n_0, n_1, ..., n_k \in \mathbf{V}_{kb}$, kde $\forall i \in 0..k \; n_i[\varphi] = \varphi_i$, $n_0 \in \mathbf{R}_{kb}$, $n_k = n$ a $\forall i \in 1..k \; \exists e_i \in \mathbf{E}_{kb}, e_i[\alpha] = n_{i-1}, e_i[\beta] = n_i, e_i[\tau] =$ "$inferior$". Úplné jméno uzlu zapisujeme tečkovou notací "$\varphi_0.\varphi_1.\varphi_2. \; ... \; .\varphi_k$", tedy například "PATIENT.PHYS_EXAM.BP.SYSTOLIC".

Úplné jméno uzlu jednoznačně identifikuje uzel znalostní báze. Důkaz provedeme indukcí. Pro n_0 předpokládáme $n_0 \in \mathbf{R}_{kb}$. Dle (6) tedy $\forall n' \in \mathbf{R}_{kb}, \; n' \neq n \Rightarrow n'[\varphi] \neq n_0[\varphi] = \varphi_0$, tedy "$\varphi_0$" jednoznačně identifikuje nějaký kořen znalostní domény. Předpokládejme nyní, že řetězec "$\varphi_0.\varphi_1.\varphi_2. \; ... \; .\varphi_i$" jednoznačně identifikuje uzel n_i. Postupujme sporem, nechť řetězec "$\varphi_0.\varphi_1.\varphi_2. \; ... \; .\varphi_i.\varphi_{i+1}$" odpovídá dvěma uzlům $n_j, n_k \in \mathbf{V}_{kb}, n_j \neq n_k$. Uzel n_i je ale jednoznačně určen posloupností "$\varphi_0.\varphi_1.\varphi_2. \; ... \; .\varphi_i$", tedy $\exists e \in \mathbf{E}_{kb}, e[\alpha] = n_i, e[\beta] = n_j, e[\tau] =$ "$inferior$". Stejně tak pro n_k máme $\exists e' \in \mathbf{E}_{kb}, e'[\alpha] = n_i, e'[\beta] = n_k, e'[\tau] =$ "$inferior$". Tedy z (4) plyne, že $n_j \diamondsuit n_k$, pak ale dle (6) musí být $\varphi_{i+1} = n_j[\varphi] \neq n_k[\varphi] = \varphi_{i+1}$, což je spor.

5. Mobilní přístup k záznamu

5.1. Využití tenkých klientů

Ze schématu architektury na obrázku 1 je patrno, že pro typickou komunikaci mezi klienty a aplikační vrstvou je využíváno XML definované v [6]. Kvůli příkazově orientovanému charakteru tohoto XML je nutné, aby standardní klientské aplikace disponovaly nezanedbatelnou výpočetní silou.

Pro přístup k záznamu z tenkých klientů je využívána množina služeb na straně aplikační vrstvy, která transformuje příkazově orientované XML do HTML či WML jazyka. Tyto utility jsou implementovány jako speciální preprocesory ve formě CGI programů či HTTP Server modulů. Tímto způsobem může být veškerá složitost a aplikační logika přesunuta na prostřední vrstvu MUDR záznamu, což zjednoduší výpočetní nároky na uživatelské rozhraní a umožní využít klienty ve formě WWW prohlížečů, Pocket, Handheld či Tablet PC, PDA nebo mobilních telefonů. Jedním z prvních testovacích modulů byl modul pro Nokia 9110i Communicator. U tohoto zařízení bylo nutné pamatovat na to, že WWW prohlížeč nepodporuje ani tabulky ani rámce. Obecně je velmi důležité přizpůsobit výstup podmínkám malého přenosného zařízení. V porovnání z osobními počítači je třeba počítat s malým a často monochromatickým displejem, omezenými možnostmi ovládání, menší pamětí a výpočetní sílou, pomalejším datovým přenosem atd. S přihlédnutím k těmto omezením zvažujeme řešení, které by umožňovalo předtřídit data dle specifikace lékaře a znalosti informací o pacientovi tak, aby pouze relevantní data byla přenášena k lékaři a zobrazována na mobilním zařízení. Ostatní data by byla přístupná až na speciální vyžádání. Tímto by došlo ke zmenšení objemu přenášených dat a snížení potřebné přenosové kapacity. Bohužel, tento způsob stále naráží na obavy z možných následků chybného rozhodnutí na základě neúplné informace.

Jak je vidět ze schématu na obrázku 2, v nové verzi je použití tenkých klientů přesunuto až za tzv. "MUDR WS Proxy" službu. Je tomu tak proto, že využití tenkých klientů je čím dál více potlačováno. Důvody k tomuto jsou dva. Příliš jednoduché zařízení nedokáže dostatečně přehledně zobrazit potřebné informace tak, aby lékaři byli ochotni s těmito informacemi pracovat. Kromě toho s vývojem v oblasti mobilních komunikací přichází na trh mnoho mobilních zařízení, která umožňují implementace tzv. tlustých mobilních klientů, kterým se ve své práci snažím věnovat více.

5.2. Tlustí, ale mobilní klienti

Nejnovější trend výzkumu mobilního přístupu k EHR MUDR spočívá v tvorbě tzv. ".NET Compact Klienta". K vývoji je používán ".NET Compact Framework" [7], který je přirozeně podporován ve vývojovém nástroji "MS Visual Studio .NET 2003" [8, 9]. Tento nástroj v kombinaci s programovacím jazykem $C\#$ přispívá ke vzniku MUDR mobilního klienta na platformách "Pocket PC" a "Smart Phone 2002". Pomocí .NET Studia rozšířeného o tzv. "Smart Device Extensions", případně v kombinaci s "Microsoft Mobile Internet Toolkit", lze v současné době jednoduše vyvíjet pro více než 200 různých zařízení, přičemž tento počet velmi rychle vzrůstá. Díky tomu, že stále více zařízení používá Windows XP Embedded and Windows CE .NET [10] operační systémy, je vývoj do jisté míry transparentní záležitostí.

V současné době jako testovací a vyvíjecí platformu pro mobilního MUDR Klienta využíváme zařízení T-Mobile MDA. Tento mobilní digitální asistent kombinuje výhody mobilního telefonu podporujícího GPRS s osobním digitálním asistentem (PDA) na platformě Pocket PC 2002 Phone Edition (WinCE 3.0). Kombinací výkonu procesoru Intel Arm SA-1110 206MHz, barevného TFT 240 x 320 pix. dotykového displeje a 32 MB RAM rozšiřitelné pomocí MMC karet je poskytována dostatečná funkčnost pro použití v medicínském prostředí.

Další vývoj směřujeme k použití tzv. ultra osobních počítačů, které představila firma Microsoft na "Windows Hardware Conference 2002" (WinHEC 2002). Tyto počítače slibují posunout éru osobních počítačů kupředu srovnatelným způsobem jako posunul příchod mobilních telefonů svět telekomunikací. První produkt této řady by měl být k dispozici koncem roku 2003. Jde o plně funkční všestranný wireless handheld počítač, který jednoduše poslouží i jako notebook. Počítač měřící cca. 10,5 cm x 7,4 cm x 2,3 cm a vážící ne více než 250 gramů využívá operační systém Microsoft Windows XP Professional, zahrnuje 1GHz Crusoe TM5800 processor od firmy Transmeta Corporation, 4 palcový barevný VGA LCD displej s dotykovou obrazovkou, USB, zvukový výstup a podporu pro bezdrátové sítě 802.11b and Bluetooth.

5.3. Komunikace mezi mobilními klienty a MUDR záznamem

Jak již bylo řečeno, tlustý mobilní klient využívá MUDRWSAPI ke komunikaci s elektronickým zdravotním záznamem MUDR. V praxi to znamená, že na straně klienta je třeba vytvořit tzv. "proxy objekt" webové služby MUDR Web Service. Tento proxy objekt nazveme např. `MUDRWSProxy` a podědíme od třídy `System.Web.Services.Protocols.SoapHttpClientProtocol`. Pro každou metodu rozhraní MUDRWSAPI jsou vytvořeny 3 metody proxy třídy, jedna stejného názvu jaký má veřejná funkce rozhraní, jedna s prefixem `Begin` a jedna s prefixem `End`. První metoda slouží pro normální synchronní vyvolání určité funkce MUDRWSAPI, další dvě jsou určeny pro asynchronní komunikaci. Ve vývojovém prostředí MS .NET Studia je tvorba proxy třídy automatizována na základě WSDL dokumentu generovaného webovou službou [5]. Máme-li k dispozici proxy třídu, lze jednoduchým kódem vzdáleně volat rozhraní MUDRWSAPI:

```
try
```

```
{
    cz.euromise.unix.MUDRWSProxy mws = new cz.euromise.unix.MUDRWSProxy();
    mws.Url = "http://unix.euromise.cz:6004/MUDRWebServices/MUDRWS1.asmx";
    KnowledgeNode nd = mws.get_knowledge_node("PATIENT.PHYS_EXAM.BP.SYSTOLIC");
}
catch (Exception ex)
{   Msg.Text = "Chyba při volání služby: " + ex.Message;  }
```

6. Závěr

Využití a možnosti mobilního přístupu k elektronickému zdravotnímu záznamu závisí podstatnou mírou na vývoji v oblasti mobilních komunikací a na výzkumu a rychlém vývoji podpůrných nástrojů pro lékaře. Praktické testování ukázalo, že na příliš malých a jednoduchých zařízeních není možné strukturovat medicínské informace tak, aby lékaři byli ochotni s elektronickým zdravotním záznamem tímto způsobem pracovat. Nicméně částečné využití se nachází i zde například v rychlém zobrazení přehledu stavu pacienta s možností online vkládání jednoduchých poznámek do zdravotního záznamu.

Na druhou stranu se v dnešní době objevuje na trhu spousta malých zařízení s relativně velkým displejem a dostatečnou silou na to, aby bylo možné implementovat plnohodnotné mobilní klienty elektronického zdravotního záznamu. Ve své práci se snažíme toto respektovat při vývoji MUDR mobilních modulů i při dalším výzkumu v oblasti telemedicíny. V oblasti mobilního přístupu k datům využíváme nové technologie a nejvíce podporované a rozšířené nástroje.

Bohužel tomu, aby mobilní přístup k elektronickému zdravotnímu záznamu přešel z testovacích laboratorních podmínek do běžné praxe každého lékaře, brání zatím dva zásadní problémy. Jedním je stále poměrně vysoká cena za přenesená data a druhým poměrně malá úspěšnost těchto zařízení při rozpoznávání rukopisu, zvláště pak jde-li o český jazyk. První problém by bylo možné řešit "cacheing technikami" v kombinaci s částečnou synchronizací dat občasným přímým propojováním s pevnou pracovní stanicí, na řešení druhého problému pracují vývojáři komerčních firem.

Poděkování

Práce je částečně podporována projektem LN00B107 Ministerstva školství, mládeže a tělovýchovy České Republiky.

Literatura

[1] Zvárová J., Hanzlíček P., Přibík V.: "Application of ORCA multimedia EPR in Czech hospitals", *Proceedings of 3rdEuropean Conference on Electronic Healthcare Records*, Sevilla 1999, ss. 160–165.

[2] CEN/TC251, ENV 13606, části 1 – 4.

[3] Špidlen J.: "Databázová reprezentace medicínských informací a lékařských doporučení", *Diplomová práce*, UK MFF, Katedra softwarového inženýrství, 2002.

[4] McLean S., Naftel J., Williams K.: "Microsoft .NET Remoting", Microsoft Press, 2002, ISBN 0-7356-1778-3.

[5] Shortm S.: "Building XML Web Services for the Microsoft .NET Platform", Microsoft Press, 2002, ISBN 0-7356-1406-7.

[6] Špidlen J., EuroMISE Centrum - Kardio: "Definice aplikačního rozhraní MUDR API", http://www.euromise.cz/MUDRAPI.xsd, 2002.

[7] Wigley A., Wheelwright S., Burbidge R., MacLeod R., Sutton M.: "Microsoft .NET Compact Framework (Core Reference)", Microsoft Press, 2003, ISBN 0-7356-1725-2.

[8] Kačmář D.: "Programujeme .NET aplikace ve Visual Studiu .NET", Computer Press, 2001, ISBN 8072265695.

[9] Prosise J.: "Programming Microsoft .NET", Microsoft Press, 2002, ISBN 0-7356-1376-1.

[10] Boling D.: "Programming Microsoft Windows CE .NET, Third Edition", Microsoft Press, 2003, ISBN 0-7356-1884-4.

Klasifikace satelitních snímků neuronovými sítěmi

doktorand:
MGR. INKA VYORÁLKOVÁ
Ústav informatiky Akademie věd české republiky,
Pod Vodárenskou věží 2, Praha 8
incav@cs.cas.cz

školitel:
DOC. ING. EMIL PELIKÁN, CSC.
Ústav informatiky Akademie věd české republiky,
Pod Vodárenskou věží 2, Praha 8
emil@cs.cas.cz

obor studia:
Kartografie a geoinformatika

Abstrakt

Práce je zaměřena na klasifikaci multispektrálních satelitních snímků metodou umělých neuronových sítí, jejíž vývoj a využívání je zatím v počátcích. Práce se soustřeďuje především na konkrétní zpracování snímku touto metodou, přičemž důležitým bodem je navržení postupu zpracování v dostupných programech. Základem je řízená klasifikace neuronovými sítěmi do několika tříd land use/land cover v programu Statistica Neural Networks se současným využitím programu Idrisi. Cílem práce bylo nejen vyzkoušet klasifikace různými neuronovými sítěmi, ale také porovnat výsledky z takto vytvořených klasifikací s obvykle používanou metodou maximum likelihood a s klasifikacemi neuronových sítí v programu Geomatica.

1. Úvod

Využití dat dálkového průzkumu Země (DPZ) je v současné době stále častější a velmi rozsáhlé. Vzhledem k nejrůznějším aplikacím využívaným například pro zemědělskou evidenci půdního fondu, mapování využití území, zjišťování zamokření, lesní hospodářství či pro analýzy životního prostředí, se objevuje i mnoho rozličných metod zpracování těchto dat. Jelikož přesnost a správnost vyhodnocení snímků závisí také na správně zvolené metodě klasifikace, není divu, že se objevují stále nové přístupy. Mezi nejnovější metody klasifikace obrazu patří klasifikace pomocí umělé neuronové sítě (Artificial Neural Network Classification).

Současně s nárůstem rozsahu dostupných dat DPZ přibývají i možnosti použití neuronových sítí, které se tak začaly používat i pro klasifikaci multispektrálních snímků. V posledních letech bylo zveřejněno několik aplikací klasifikačních přístupů založených na neuronových sítích, které ukázaly výhody umělých neuronových sítí oproti obvykle používaným statistickým klasifikátorům. Přístupy neuronových sítí jsou totiž nezávislé na statistickém rozložení dat a dokáží odhadnout nelineární vztah mezi vstupními a požadovanými výstupními daty. Nejčastěji se v této problematice využívá metoda řízené klasifikace obrazu neuronovou sítí Multi-Layer Perceptron feed-forward back-propagation, neboli vícevrstvou sítí se zpětným šířením při učení. Výzkum klasifikačních metod s neuronovými sítěmi je však stále ve vývoji, kterému by měla pomoci i tato práce.

2. Postup klasifikace

Ke zpracování zvoleného tématu byl použit výřez z multispektrálního satelitního snímku ze skeneru ETM+ družice Landsat 7 z roku 2000 s rozlišením 1 pixelu 30 m. Zvolené území v oblasti Polabí, přibližně mezi Litoměřicemi a Mělníkem, splňuje předpoklad různorodosti krajiny potřebný pro klasifikaci využití území.

Pro možnost dobrého porovnání výsledků klasifikací byla zvolena řízená klasifikace. Řízené klasifikace používají soubor dat se známými výstupy (tzv. trénovací data), na nichž se klasifikátor naučí rozpoznat jednotlivé případy. Bylo tedy nejprve nutné co nejlépe vybrat klasifikační třídy a trénovací soubor s referenčními daty.

K definici klasifikačních tříd byly nakonec vybrány dva trénovací soubory - pro klasifikaci do 11 a 7 tříd, jejichž výčet vidíte v obr. 1. První klasifikace do 11 tříd předpokládá velkou úspěšnost, protože jsou třídy a trénovací data dobře vybraná. Při druhé klasifikaci do 7 tříd již nejsou trénovací data tak kvalitní a tak by měly být patrné rozdíly mezi jednotlivými klasifikátory.

11 TŘÍD		7 TŘÍD	
1	vodní plochy	1	vodní plochy
2	jehličnaté lesy	2	jehličnaté lesy
3	listnaté lesy	3	listnaté lesy
4	urbanizované, zastavěné plochy	4	urbanizované, zastavěné plochy
5	holá půda, dopravní stavby apod.	5	holá půda, dopravní stavby apod.
6	zemědělské plochy 1	6	louky
7	zemědělské plochy 2	7	zemědělské plochy
8	zemědělské plochy 3		
9	zemědělské plochy 4		
10	zemědělské plochy 5		
11	zemědělské plochy 6		

Obrázek 1: Klasifikační třídy.

Po vytvoření trénovacího souboru bylo možné snímek klasifikovat do zvolených tříd. Základem byla samozřejmě klasifikace neuronovými sítěmi v programu Statistica Neural Networks (SNN), kde jsem vyzkoušela různé typy a architektury neuronových sítí. Některé takto vytvořené klasifikace jsem pak porovnala se stejnými, vytvořenými v programu Geomatica, tedy v programu pro zpracování dat DPZ s nově zabudovaným modulem pro klasifikaci neuronovými sítěmi.

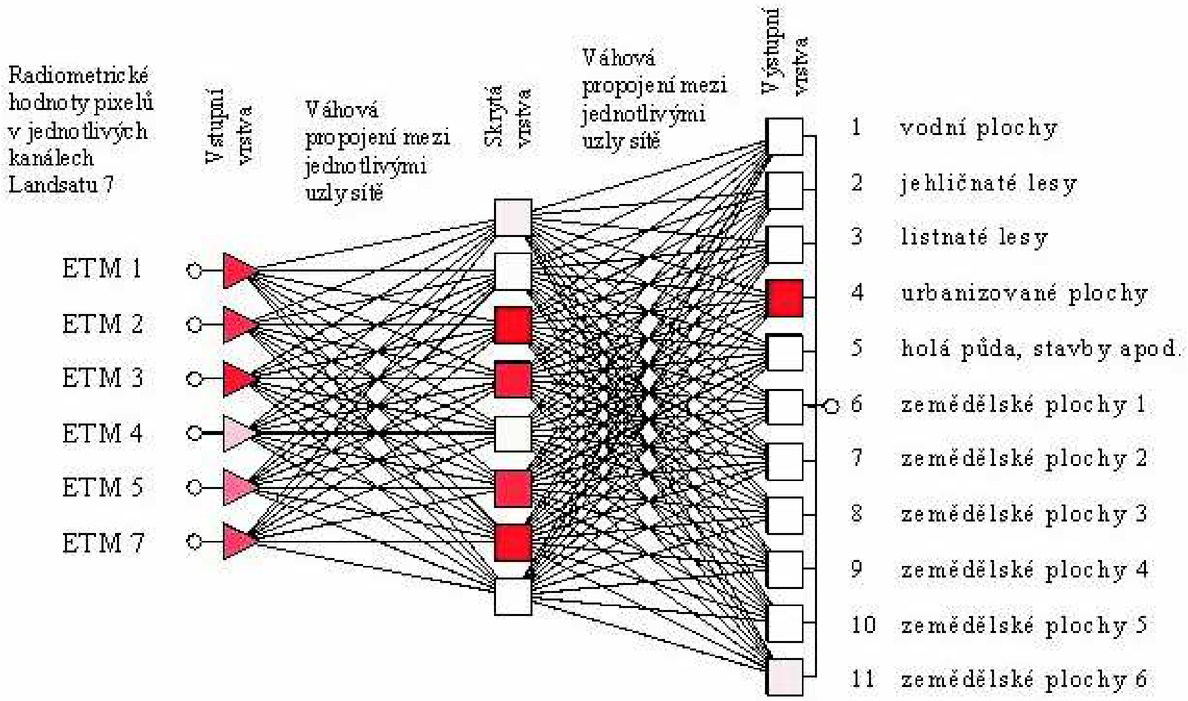
Zda jsou klasifikátory neuronových sítí skutečně lepší než běžně používané klasifikace bylo možné posoudit až po vytvoření klasifikací maximum likelihood. Tyto klasifikátory založené na pravidle největší pravděpodobnosti podávají ze statistických klasifikátorů většinou nejlepší výsledky. Srovnávací klasifikace maximum likelihood byla vytvořena v programech Idrisi a Geomatica.

3. Klasifikace neuronovými sítěmi

Ke zkoumání možností klasifikace snímků neuronovými sítěmi musel být navržen takový postup, který bude využívat profesionální program určený k řešení úloh pomocí neuronových sítí. Tomuto účelu plně vyhovoval program Statistica Neural Networks. K převodu snímků do tohoto programu byl použit program Idrisi, protože SNN umožňuje práci pouze s tabulkovými daty. Vhodnost zvoleného postupu byla posouzena srovnáním s klasifikacemi v programu Geomatica.

V programu SNN byly vyzkoušeny s různým nastavením čtyři záladní typy neuronových sítí- Multilayer Perceptron, Radial Basis Function, Lineární a Pravděpodobnostní neuronové sítě. Nejlepší výsledky dávala klasifikace neuronovou sítí Multilayer Perceptron s trénováním backpropagation, proto bude blíže vysvětlen především princip této klasifikace. Vstupem do klasifikace jsou radiometrické hodnoty z jednotlivých kanálů Landsatu 7, které odpovídají odrazivosti objektů v jednotlivých spektrálních pásmech a určují velikost vstupního signálu. Vnitřní parametry sítě pak musí být nastaveny tak, aby největší výstupní signál byl u neuronu odpovídajícího požadované třídě. Toto nastavení se upravuje během trénování neuronové sítě a záleží na něm konečná přesnost klasifikace.

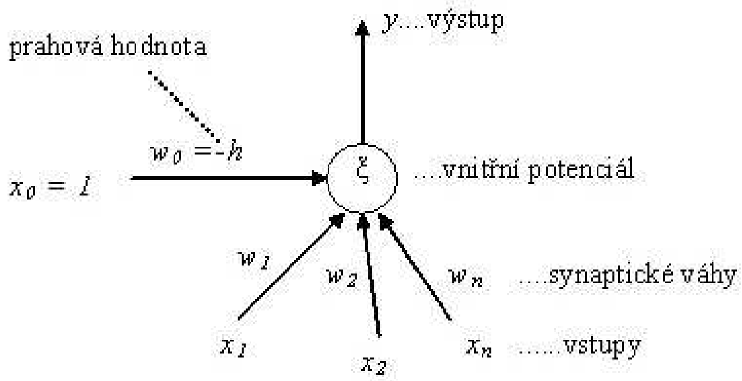
Příklad zpracování snímku je znázorněn na obrázku 2 s vyznačenými aktivacemi neuronů pro pixel z městské zástavby, který má téměř ve všech kanálech vysokou odrazivost. Nastavená váhová propojení pak tyto vstupní signály změní tak, že nejvyšší výstupní signál vyjde právě u 4. výstupního neuronu. Ke všem ostatním výstupním neuronům váhy tento signál utlumí, kdežto směrem k požadovanému uzlu ho zvýší. Takto se pak při klasifikaci zpracuje pixel po pixelu celý obraz.



Obrázek 2: Příklad třívrstvé perceptronové sítě se 6 vstupními, 8 skrytými a 11 výstupními uzly (MLP 6-8-11) s příkladem vstupu a výstupu při klasifikaci satelitních snímků (s vyznačenými aktivacemi pro město).

Neurony v síti jsou propojeny tzv. váhovými propojeními, které zesilují nebo zeslabují signál přicházející z předchozích neuronů. Suma těchto vážených signálů určuje aktivaci neuronu, která ovlivňuje další výstup z neuronu (viz (1). Výstup z neuronu je pak funkcí právě této aktivace, jak je možné vidět na obr. 3. a ve vzorci (2), kde je výstup vypočten na základě nejčastější logistické aktivační funkce.

$$aktivace\ \ neuronu\ \ a = \sum_{i=1}^{n} w_i x_i - h \qquad (1)$$



Obrázek 3: Matematický model neuronu.

výstup y:

$$y = f(a) = \frac{1}{1 + e^{-a}} \tag{2}$$

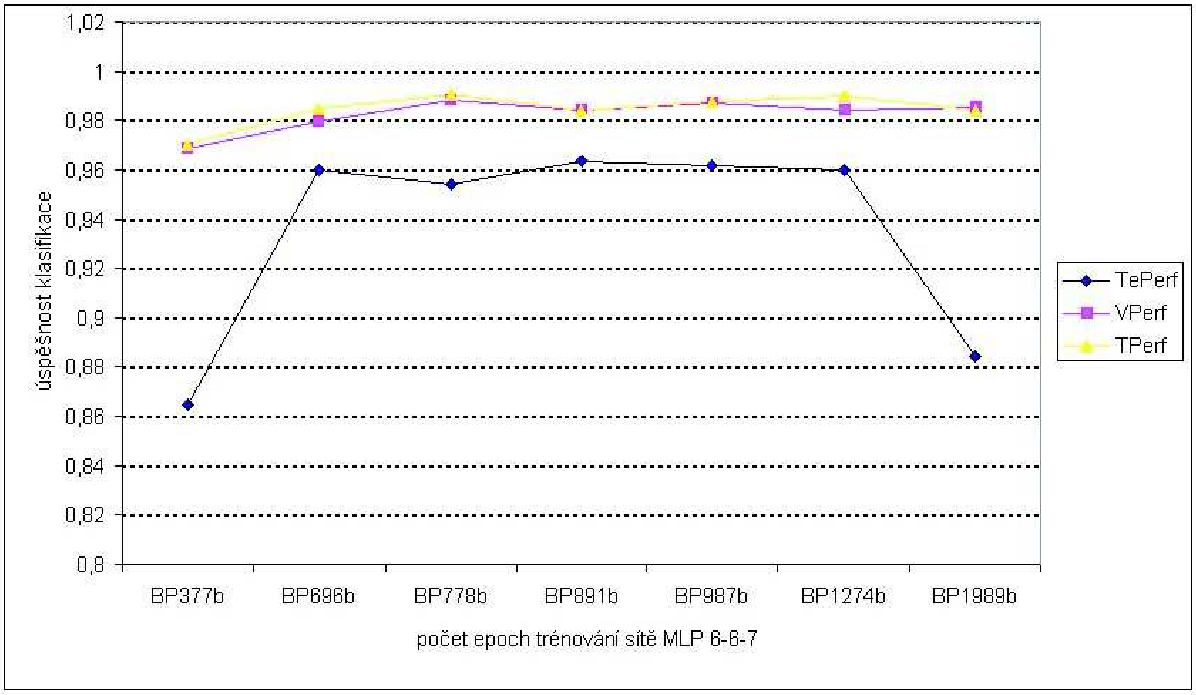
Nastavení vah tedy hraje ve fungování neuronové sítě důležitou roli. Ke správnému nastavení dojde při úspěšném trénování neuronové sítě, k čemuž použijeme trénovací soubor, ve kterém známe správné zařazení pixelů do jednotlivých klasifikačních tříd. Při trénování neuronové sítě se § signály nejprve vyšlou směrem dopředu, u výstupních neuronů se porovnají výstupy s požadovanými a zjištěné chyby se použijí ke změně nastavení vah v síti. Hledání vhodných parametrů sítě se opakuje dokud nejsou chyby mezi požadovaným a aktuálním výstupem minimální. Tímto způsobem je neuronová síť schopna rozpoznat i statisticky podobná data a proto je úspěšnost její klasifikace velmi vysoká.

4. Využití programu Statistica Neural Neworks

Statistica Neural Networks (SNN) je produktem americké firmy StatSoft, Inc., Tulsa (http://www.statsoft.com). Tento profesionální program napsaný v jazyce C/C++ je samostatně fungující částí softwaru Statistica určenou k řešení úloh pomocí neuronových sítí. V práci byla použita dostupná verze programu SNN 4.0 E z roku 2000. Program pracuje s daty v tabulkové podobě a umožňuje snadný převod do jiných aplikací.

SNN podporuje práci s šesti základními typy neuronových sítí - Multilayer Perceptrons, Radial Basis Functions, Kohonen, Probabilistic, Generalized Regression a Principal Components. V tomto programu tedy můžeme vytvářet různé typy a architektury neuronových sítí, které můžeme učit různými trénovacími algoritmy na vzorových datech. Natrénované neuronové sítě pak mohou být užity k řešení regresních či klasifikačních úloh, jejichž řešení jinými metodami selhává.

Základní výhodou programu SNN je možnost automatického hledání nejvhodnější neuronové sítě. Toto hledání může trvat až 3 dny (na počítači s procesorem 500 MHz), ale vyzkouší všechny přijatelné neuronové sítě a určí jejich přesnosti na zadaných datech. V tomto programu také můžeme sledovat proces trénování neuronové sítě na grafu, který ukazuje snižování chyby na trénovacích a ověřovacích datech. Začnou-li se tyto křivky oddalovat, je vhodné proces trénování ukončit. Uvedené možnosti však například v programu Geomatica, který umožňuje při klasifikaci neuronovými sítěmi pracovat přímo s obrazovými daty, chybí. Výhody možnosti ovlivňování trénovacího algoritmu lze sledovat na obr. 4. Na tomto příkladu je vidět, že při včasném zastavení trénování bude neuronová síť schopna dobře rozpoznat i neznámá data.



Obrázek 4: Vliv počtu epoch na úspěšnost trénování. ((T/V/Te)Perf=přesnost klasifikace pro trénovací/ověřovací/testovací data, BP250=trénování back-propagation se 250 epochami).

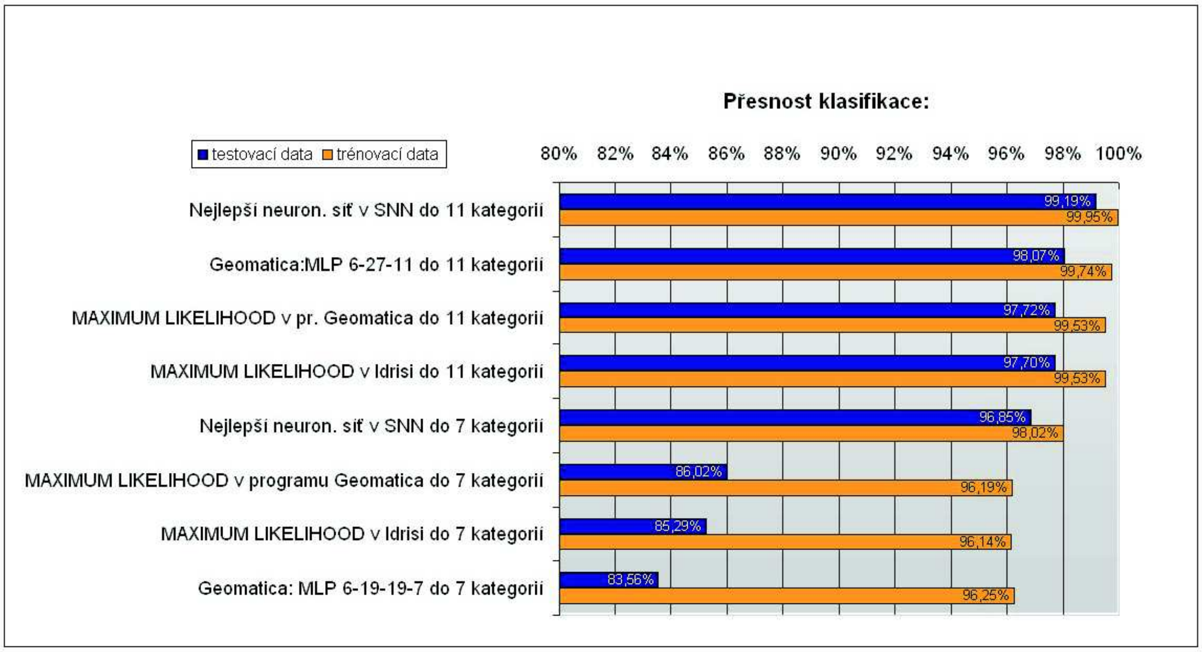
5. Hodnocení výsledků klasifikace

Vyhodnocení úspěšnosti klasifikace je kontrolou přesnosti celého klasifikačního procesu. Přesnost klasifikace má určit nakolik odpovídá vytvořená tématická mapa skutečnosti, k čemuž se používají tzv. testovací území, u nichž známe správné zařazení, ale která nejsou shodná s trénovacími daty. V následujících tabulkách a grafu (obr. 5 - 7) je možné vidět, jak se přesnosti jednotlivých klasifikací od sebe lišily.

Při klasifikaci do 11 tříd se jednotlivé klasifikátory skutečně příliš nelišily. Nejlepší třívrstvá neuronová síť v programu SNN klasifikovala téměř neuvěřitelně na 99,19 % správně. Stejná síť v programu Geomatica však klasifikovala o 1 % hůře a klasifikátory největší pravděpodobnosti jen s nepatrně horšími výsledky. Přesnosti klasifikací do 7 tříd se již liší více. Klasifikace v programu SNN byly přibližně o 10 % lepší než všechny ostatní.

Porovnání výsledků klasifikací lze shrnout do následujících bodů:

- Neuronové sítě klasifikují přesněji než maximum likelihood, což byl předpokládaný výsledek, ale bylo nutné ho dokázat.
- Navržený postup klasifikace v programu Statistica Neural Networks dává lepší výsledky než stejné klasifikace v programu Geomatica. Tato skutečnost může být způsobena nevhodným trénovacím algoritmem nebo chybějící možností hledání nejvhodnější neuronové sítě v programu Geomatica.
- Dalším vpodstatě očekávaným zjištěním bylo to, že nejlepšími neuronovými sítěmi ke klasifikaci dat DPZ jsou vícevrstvé perceptronové sítě, i když sítě typu RBF nebo pravděpodobnostní sítě dávají přijatené výsledky. Lineární neuronové sítě jsou k tomuto účelu nevhodné.
- Výsledky klasifikace jsou pak samozřejmě vždy lepší při lépe vybraném trénovacím souboru, ale tento rozdíl je znatelný především u statistického klasifikátoru. Neuronové sítě totiž nejsou závislé na statistickém rozložení dat.



Obrázek 5: Srovnání výsledků klasifikací.

6. Výstupy z klasifikace

Definitivní výstupy v podobě digitálních rastrových map land use/land cover jsou vytvořeny z nejlepších klasifikací neuronovými sítěmi po vyhlazení obrazu a připojení na souřadnice S-42, příklad je uveden na obr. 8. Tyto výstupy mohou sloužit pro určení využití konkrétního území podle souřadnic, nebo mohou být vstupem do geografických informačních systémů (GIS) v rastrové, případně i vektorové podobě (po vytvoření vrstev pro jednotlivé kategorie).

7. Závěr

Navržený postup klasifikace snímků v programu Statistica Neural Networks podává lepší výsledky než běžně používané klasifikace a tudíž může být doporučen dalším uživatelům. Tím se rozšiřují možnosti využití tohoto moderního klasifikátoru, protože v dosavadních studiích si autoři většinou vytvářeli vlastní programy, které jsou nedostupné. Až v posledních letech se začaly objevovat nové moduly s tímto klasifikátorem v programech specializovaných na zpracování dat DPZ, ty jsou však obvykle velmi drahé.

Kromě možnosti dalšího využití výsledků klasifikace přináší napsaná práce také mnoho námětů na další rozšiřování zpracovaného tématu. Zajímavé by bylo například vyzkoušet klasifikace neuronovými sítěmi s plynulými přechody mezi třídami, zahrnout do klasifikace další vstupní data (například výšky) nebo zohlednit informace o okolí pixelu. Dále by také bylo možné zabývat se příčinami nespolehlivosti klasifikací neuronovými sítěmi v programu Geomatica. Budoucí práce bude tedy zaměřena především na zahrnutí okolí pixelu a dalších vstupních dat do podrobnějších klasifikací.

KLASIFIKACE NEURONOVÝMI SÍTĚMI V SNN - TYP SÍTĚ:			PŘESNOST KLASIFIKACE		
Třívrstvé neuronové sítě typu Multilayer Perceptron (MLP)			trénovací data	testovací data	Poř.
37	**MLP 6-11-7**	Trénování BP50,CG201	99.2667%	96.1050%	14
38	**MLP 6-33-7**	Trénování BP50,CG186	99.3258%	95.9659%	17
39	**MLP 6-43-7**	Trénování BP50,CG224	99.4636%	96.3137%	9
81	**MLP 6-6-7**	Trénování BP891	98.4178%	96.3311%	8
68	**MLP 6-6-7**	Trénování BP363	97.9814%	95.5486%	19
87	**MLP 6-19-7**	Trénování BP866 ,CG21	99.1476%	96.2441%	13
88	**MLP 6-19-7**	Trénování BP1859	99.2658%	96.3137%	10
92	**MLP 6-19-7**	Trénování BP980 ,CG47	99.2658%	96.3137%	11
Čtyřvrsvé neuronové sítě typu Multilayer Perceptron (MLP)					
17	**MLP 6-16-12-7**	Trénování BP50,CG165	99.1671%	96.5745%	3
19	**MLP 6-16-18-7**	Trénování BP50,CG58	99.1667%	96.4702%	4
33	**MLP 6-37-29-7**	Trénování BP50,CG120	99.1871%	96.3659%	7
34	**MLP 6-40-36-7**	Trénování BP50,CG131	99.3449%	96.2963%	12
108	**MLP 6-19-19-7**	**Trénování BP33**	**98.0218%**	**96.8527%**	**1**
112	**MLP 6-19-19-7**	Trénování BP86,CG18	99.0489%	96.3832%	5
115	**MLP 6-19-19-7**	Trénování BP33,CG74	99.1476%	96.5919%	2
118	**MLP 6-18-18-7**	Trénování BP76	98.2987%	96.3832%	6
124	**MLP 6-18-21-7**	Trénování BP35	97.5672%	95.4964%	20
126	**MLP 6-20-20-7**	Trénování BP37	98.3952%	96.0703%	16
Pravděpodobnostní model neuronové sítě (PNN)					
24	**PNN 6-2557-7**	Prahová hodnota 512	99.2610%	04.9591%	20
25	**PNN 6-2557-7**	Prahová hodnota 434.2928	99.2618%	85.0287%	27
27	**PNN 6-2557-7**	Prahová hodnota 306.2999	99.2422%	85.1678%	26
Neuronové sítě typu Radial Basis Function (RBF)					
20	**RBF 6-339-7**	Trénování KM,KN,PI	98.8498%	95.8964%	18
21	**RBF 6-339-7**	Trénování KM,KN,PI	98.7903%	96.0876%	15
KLASIFIKACE NEURONOVÝMI SÍTĚMI v programu Geomatica					
MLP 6-19-19-7		Trénování BP10	96.2483%	83.5576%	29
MLP 6-19-19-7		Trénování BP22	97.2461%	83.3479%	30
MLP 6-19-19-7		Trénování BP33	96.6873%	83.1207%	31
MLP 6-19-19-7		Trénování BP66	97.2860%	77.5293%	33
MLP 6-6-7		Trénování BP72	97.0465%	88.9394%	21
MLP 6-6-7		Trénování BP234	97.1662%	87.5415%	22
MLP 6-6-7		Trénování BP363	97.2461%	86.7901%	23
MLP 6-20-20-7		Trénování BP37	95.9689%	79.6610%	32
KLASIFIKACE MAXIMUM LIKELIHOOD v Idrisi			96.1440%	85.2895%	25
KLASIFIKACE MAXIMUM LIKELIHOOD v programu Geomatica			96.1884%	86.0213%	24

BP250=trénování back-propagation se 250 epochami,
CG=trénování conjugate gradients,
KM=K-Means (Center Assignment),
KN=K-Nearest Neighbour (Deviation Assignment),
PI=Pseudo-Invert (Linear Least Squares Optimization)
Poř. = Pořadí podle přesnosti na testovacích datech

MLP=Multilayer Perceptron,
RBF=Radial Basis Function,
PNN=Probabilistic Neural Network

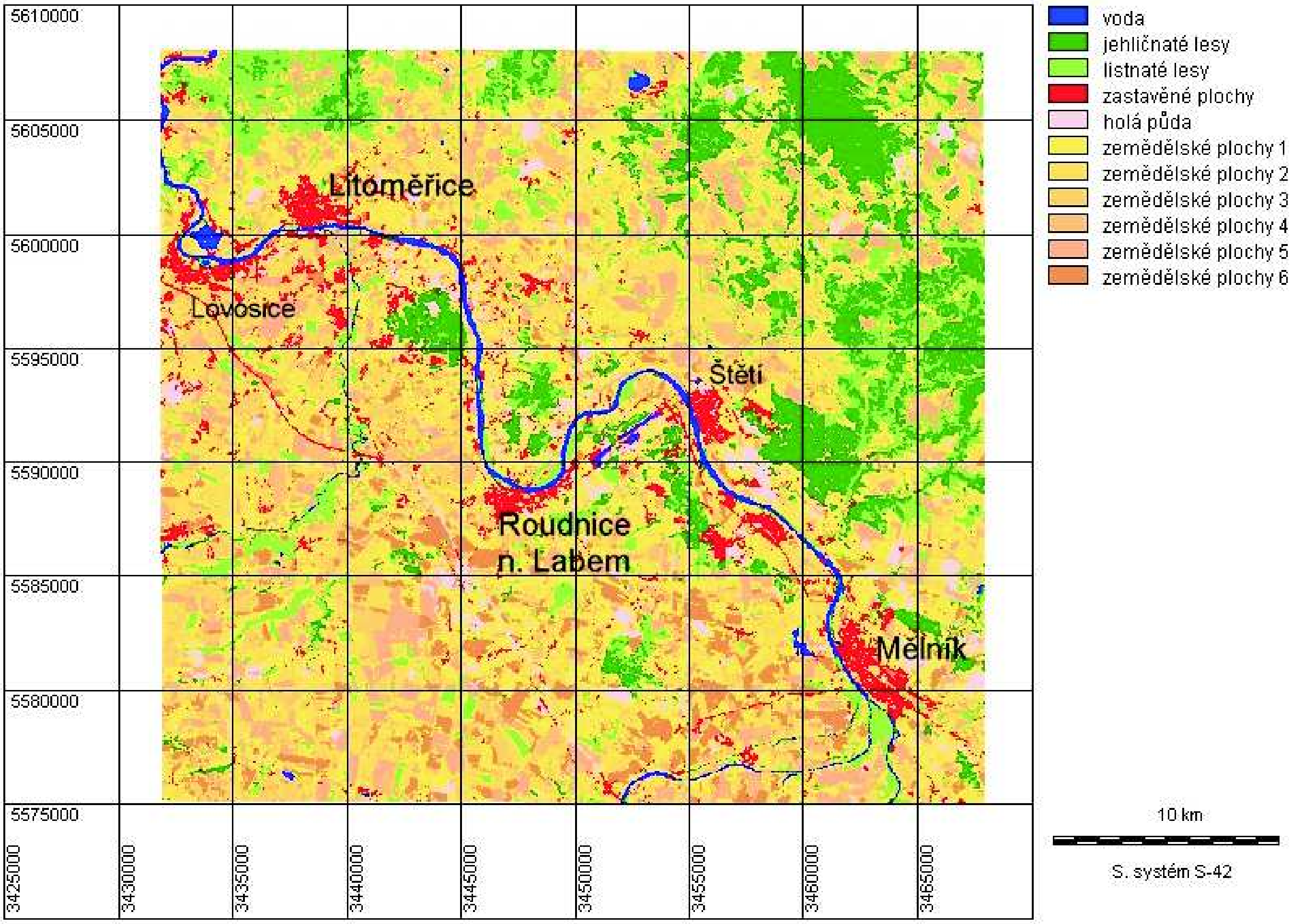
Obrázek 6: Přehled celkových přesností jednotlivých klasifikací do 7 tříd.

KLASIFIKACE NEURONOVÝMI SÍTĚMI V SNN - TYP SÍTĚ:			PŘESNOST KLASIFIKACE		
Třívrstvé neuronové sítě typu Multilayer Perceptron (MLP)			trénovací data	testovací data	Poř.
5. MLP 6:6-27-11:1	**Trénování**	**BP50,CG123**	**99.94779%**	**99.19040%**	**1**
10. MLP 6:6-24-11:1	Trénování	BP50,CG124	99.94779%	99.08920%	2
6. MLP 6:6-28-11:1	Trénování	BP50,CG84	99.92169%	98.81451%	3
12. MLP 6:6-16-11:1	Trénování	BP50,CG143	99.94779%	98.66994%	5
28. MLP 6:6-8-11:1	Trénování	BP239	99.60846%	98.33743%	12
Čtyřvrstvé neuronové sítě typu Multilayer Perceptron (MLP)					
1. MLP 6:6-44-44-11:1	Trénování	BP50,CG56	99.92169%	98.62657%	6
22. MLP 6:6-15-20-11:1	Trénování	BP50,CG170	99.92169%	98.53983%	8
9. MLP 6:6-29-29-11:1	Trénování	BP50,CG135	99.92169%	98.33743%	11
25. MLP 6:6-29-29-11:1	Trénování	BP50,CG50,CG0	99.89559%	98.56874%	7
24. MLP 6:6-17-18-11:1	Trénování	BP50,CG50,CG71	99.86949%	98.75669%	4
Pravděpodobnostní model neuronové sítě (PNN)					
13. PNN 6:6-1931-11:1	Prahová hodnota	512	99.94779%	98.35189%	10
15. PNN 6:6-1931-11:1	Prahová hodnota	434.2928	99.94779%	98.35189%	9
Neuronové sítě typu Radial Basis Function (RBF)					
18. RBF 6:6-29-11:1	Trénování	KM,KN,PI	99.60846%	97.71577%	16
20. RBF 6:6-30-11:1	Trénování	KM,KN,PI	99.58235%	97.58566%	18
Lineární architektura neuronové sítě					
27. Linear 6:6-11:1	Trénování	PI	82.69381%	83.24418%	20
KLASIFIKACE NEURONOVÝMI SÍTĚMI v programu Geomatica					
MLP 6-8-11	Trénování	BP239	99.65870%	97.51994%	19
MLP 6-27-11	Trénování	BP163	99.73746%	98.07107%	13
MLP 6-27-11	Trénování	BP481	99.76372%	97.73749%	14
KLASIFIKACE MAXIMUM LIKELIHOOD v Idrisi			99.53015%	97.70132%	17
KLASIFIKACE MAXIMUM LIKELIHOOD v programu Geomatica			99.52744%	97.72299%	15

BP250=trénování back-propagation se 250 epochami,
CG=trénování conjugate gradients,
KM=K-Means (Center Assignment),
KN=K-Nearest Neighbour (Deviation Assignment),
PI=Pseudo-Invert (Linear Least Squares Optimization)
Poř. = Pořadí podle přesnosti na testovacích datech

MLP=Multilayer Perceptron,
RBF=Radial Basis Function,
PNN=Probabilistic Neural Network

Obrázek 7: Přehled celkových přesnostíjednotlivých klasifikací do 11 tříd.



Obrázek 8: Výstup z nejlepší klasifikace do 11 kategorií.

References

[1] Atkinson, P. M. - Tatnall, R. L. Neural networks in remote sensing. *International Journal of Remote Sensing*, 1997, roč. 18, č. 4, s. 699-709.

[2] Bhattacharya, U. - Parui, S. K. An improved backpropagation neural network for detection of road-like features in satellite imagery. *International Journal of Remote Sensing*, 1997, roč. 18, č. 16, s. 3337-3394.

[3] Bischof, H. - Schneider, W. - Pinz, A. J. Multispectral Classification of Landsat-Images Using Neural Networks. *IEEE Transactions On Geoscience And Remote Sensing*, 1992, roč. 30, č. 3, s. 482-490.

[4] Carpenter, G. A. a kol. ART Neural Networks for Remote Sensing: Vegetation Classification from Landsat TM and Terrain Data. *IEEE Transactions On Geoscience And Remote Sensing*, 1997, roč. 35, č. 2, s. 308-325.

[5] Doubrava, P. Application of Likelihood Classification for Comparative Data Testing From Scanners MSU-E and Thematic Mapper. *Kartografie a geoinformatika*, 2000, roč. 2, č. 1, s. 25-43.

[6] Doubrava, P. Klasifikace dat dálkového průzkumu pro Národní atlas ČR. *Kartografie a geoinformatika*, 1999, roč. 1, č. 1, s. 7-28.

[7] Fanta, Jiří. Neuronové sítě ve společenských vědách. Praha: *Nakladatelství Karolinum*, 2000. 165 s. ISBN 80-246-0175-3.

[8] Foody, G. M. - Arora, M. K. An evaluation of some factors affecting the accuracy of classification by an artificial neural network. *International Journal of Remote Sensing*, 1997, roč. 18, č. 4, s. 799-810.

[9] Foody, G. M. - McCulloch, M. B. - Yates, W. B. Classification of Remotely Sensed Data by an Artificial Neural Network: Issues Related to Training Data Characteristics. *Photogrammetric Engineering & Remote Sensing*, 1995, roč. 61, č. 4, s. 391-401.

[10] Frizzelle, Brian G. - Moody, Aaron. Mapping Continuous Distributions of Land Cover: A Comparison of Maximum-Likelihood Estimation and Artificial Neural Networks. *Photogrammetric Engineering & Remote Sensing*, 2001, roč. 67, č. 6, s. 693-705.

[11] Hakl, F. - Holeňa, M. úvod do teorie neuronových sítí. 1. vyd. Praha: *Vydavatelství ČVUT*, 1998. 210 s. ISBN 80-01-01716-8.

[12] Heermann, P. D. - Khazenie, N. Classification of Multispectral Remote Sensing Data Using a Back-Propagation Neural Network. *IEEE Transactions On Geoscience And Remote Sensing*, 1992, roč. 30, č. 1, s. 81-88.

[13] Hojovec, V. a kol. Kartografie. 1. vyd. Praha: *Geodeický a kartografický podnik v Praze, n. p.*, 1987. 660 s.

[14] Kanelopulos, I. - Wilkinson, G. G. Strategies and best practice for neural network image classification. *International Journal of Remote Sensing*, 1997, roč. 18, č. 4, s. 711-725.

[15] Lillesand, Thomas M. - Kiefer, Ralph W. Remote sensing and image interpretation. 3. vyd. New York: *John Wiley & Sons, Inc.*, 1994. 750 s. ISBN 0-471-57783-9.

[16] Novák, Mirko a kol. Umělé neuronové sítě: teorie a aplikace. 1. vyd. Praha: *C. H. Beck*, 1998. 382 s. ISBN 80-7179-132-6.

[17] Statistica Neural Networks. *StatSoft*, 1998. 318 s. ISBN 1-884233-42-2.

[18] šíma, J. - Neruda, R. Teoretické otázky neuronových sítí. 1. vyd. Praha: *Matfyzpress*, 1996. 390 s. ISBN 80-85863-18-9.

[19] Tomoji Yoshida - Sigeru Omatu. Neural Network Approach to Land Cover Mapping. *IEEE Transactions On Geoscience And Remote Sensing*, 1994, roč. 32, č. 5, s. 1103-1109.

Aplikace analýzy biosignálů srdce

doktorand:
ING. PETR ZAVADIL

EuroMISE
Ústav informatiky AV ČR
Pod Vodárenskou věží 2
Praha 8

zavadil@euromise.cz

školitel:
DOC. ING. VLADIMÍR ECK, CSC.

Katedra kybernetiky
FEL ČVUT v Praze
Technická 2
Praha 6

eck@lab.felk.cvut.cz

obor studia:
Umělá inteligence a biokybernetika

Abstrakt

Článek je zaměřen na vybrané metody zpracování biosignálů a jejich aplikaci na signály srdce, a to především v oblasti elektrokardiogramů (EKG). Analýza biosignálů je ilustrována na diagnostických metodách kardiografie. Doplněny jsou ukázky využití záznamu a zpracování EKG z lékařské praxe.

1. Úvod

Zpracování biosignálů představuje propojení technického přístupu zpracování signálů a diagnostických potřeb vyplývajících z lékařské praxe. Následující metodologie je vybrána z hlediska dvou požadovaných cílů:

- nejlépe zobrazit požadovaný signál,
- nalézt a zobrazit diagnosticky významné parametry signálů.

Konkrétní použití je zaměřeno na biosignály srdce, především na elektrokardiogram (EKG). Problematika EKG úzce souvisí se zaměřením dvou lékařských pracovišť, pro něž je tato analýza biosignálů primárně určena v rámci podpory či spolupráce:

- kardiologická ambulance EuroMISE centra - Kardio,
- II. interní klinika kardiologie a angiologie Všeobecné fakultní nemocnice v Praze a 1. lékařské fakulty Univerzity Karlovy

2. Analýza biosignálů

Před vlastní analýzou je většinou potřeba s biosignálem provést řadu úkonů a operací, které můžeme shrnout do *předzpracování* [1, 2]. Tato fáze se může s analýzou překrývat a zařazujeme sem např. :

- segmentaci - odělení informačních úseků z dlouhodobého signálu,
- průměrování - přispívá k potlačení šumu v signálu,
- filtraci - časová, kmitočtová, korelační, ... ,
- interpolaci - prokládání funkce měřenými hodnotami signálu,
- aproximaci - hledání aproximační funkce při splnění zadaných kritérií, nejčastěji minimalizace kvadratické odchylky.

V případě digitálního záznamu do předzpracování zahrnujeme také:

- vzorkování,
- kvantování.

Většina těchto operací bývá v praxi zabezpečena firemním softwarem, či hardwarem s poměrně omezenou možností modifikace vstupních parametrů. Záznam signálu je v lepším případě uložen v databázi a následná sofistikovanější analýza mimo firemní systém je umožněna exportem dat.

2.1. Analýza v časové oblasti

Základem analýzy biosignálů v časové oblasti je znalost jejich významných grafoelementů [3]. U signálů s repetičním či periodickým charakterem bývá často vyhodnocován průměr signálu za několik period.

Ukázka hodnocení křivky EKG, u které hraje důležitou roli amplituda, tvar a časová poloha vln, je v tabulce 1.

Tabulka 1: Ukázka grafoelementů pro hodnocení křivky EKG (charakter S-T segmentu a tvar vlny T)

S - T SEGMENT		VLNA T
POLOHA	TVAR	
základní	normální	pozitivní
	konkávní (vydutý)	negativní
zvýšená	konvexní (vypuklý)	zvětšená
	plochý	nízká
snížená	šikmý	hluboká

2.2. Analýza v kmitočtové oblasti

Pro kvalitnější vyhodnocení biosignálů a případnou detekci a odstranění šumu se používá řada transformací [4], základní transformace pro posouzení signálu v kmitočtové oblasti je Fourierova nebo kosinová transformace (se zároveň nejlepší fyzikální interpretací).

$$F(\omega) = \int_{-\infty}^{+\infty} f(t)e^{-j\omega t}dt \tag{1}$$

Vzhledem k digitalizované podobě biosignálů přecházíme od *spojité Fourierovy transformace* (1) k *Fourierově transformaci diskrétní* (2).

$$F(k) = \sum_{n=0}^{N-1} f(n)e^{-j\frac{2\pi}{N}nk}, \qquad k = 0, 1, 2, ..., N-1 \tag{2}$$

V praxi se nejčastěji používá *rychlá Fourierova transformace* (3) a (4), která například umožňuje filtraci biosignálů v reálném čase.

$$W_N = e^{-j\frac{2\pi}{N}}, \tag{3}$$

$$F(k) = \sum_{n=0}^{N-1} f(n)W_N^{nk}, \qquad k = 0, 1, 2, ..., N-1 \tag{4}$$

Rychlá Fourierova transformace vychází z předpokladu délky transformace: $N = 2^m$, využívá vlastnosti *otáčecího činitele* W_N, redukuje počet operací pro N bodů na $\frac{N}{2}log_2 N = \frac{Nm}{2}$.

2.3. Kombinovaná analýza v kmitočtové i časové oblasti

Při zpracování biosignálů naražíme na potřebu podrobněji sledovat frekvenční charakteristiku v čase: při filtraci šumu v případě, kdy se spektra signálu a šumu překrývají, nebo při sledování frekvenční charakteristiky jen vybrané části signálu [4, 3].
Časově-frekvenční analýza vychází ze dvou hlavních skupin transformací:

- lineární a
- kvadratické.

Ukažme si dvě nejrozšířenější lineární transformace.

Krátkodobá Fourierova transformace je Fourierova transformace aplikovaná na krátký úsek signálu. V rovnici (5) je uvedena pro spojitý signál.

$$STFT(\tau, \omega) = \int_{-\infty}^{+\infty} f(t)g(t-\tau)e^{-j\omega t}dt, \tag{5}$$

kde $f(t)$ je zpracovávaný signál a $g(t)$ časové okénko určující interval pro analýzu. *Diskrétní krátkodobá Fourierova transformace* je uvedena v rovnici (6).

$$F(m, \omega) = \sum_{k=-\infty}^{+\infty} f(k)g(m-k)e^{-j\omega k}dt, \tag{6}$$

kde $g(m)$ jsou časové vzorky vybraného okénka.

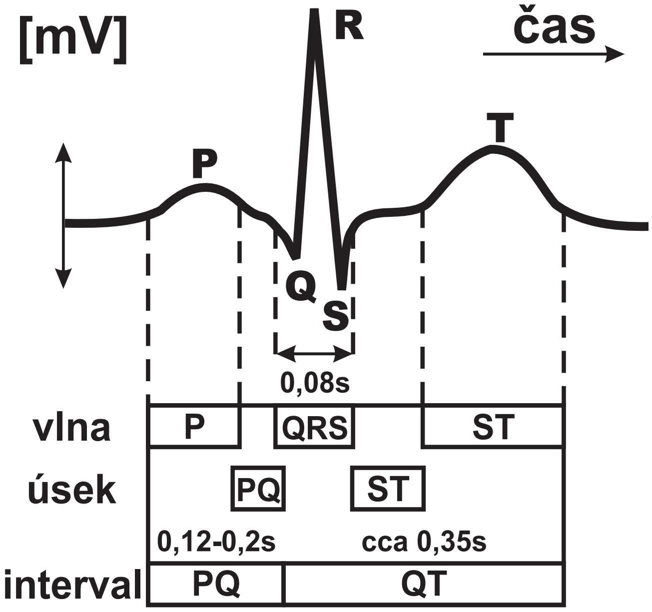
Velkou kmitočtovou rozlišovací schopnost má Vlnková transformace (Wavwlet transform) (7).

$$CWT_x(\tau, a) = \frac{1}{\sqrt{a}} \int f(t)g^* \left[\frac{t-\tau}{a}\right] dt, \tag{7}$$

kde $g(t)$ je bázová vlnka s úzkým kmitočtovým pásmem, g^* je komplexně sdružená funkce s funkcí $g(t)$ a a je rozlišovací konstanta.

3. Zpracování EKG

Elektrokardiografie podrobně mapuje signály EKG, jejich podrobnou genezi, fyziologickou souvislost a má řadu metod pro jejich vyhodnocení [5]. Základní charakteristika typického (idealizovaného) signálu EKG je na obr. 1.



Obrázek 1: Analýza EKG v časové oblasti

Ve vybraném segmentu EKG se jeho elementy rozdělují na:

- elementy náležející izolinii a
- elementy vln, komplexů a dalších grafoelementů.

Vypočítávají se křivosti oblouků, trvání vln a komplexů, velikost amplitud apod. Na základě vhodně zvolených příznaků se následně provede *třídění do určitých diagnostických tříd*.

Do oblasti časově-frekvenční analýzy EKG spadá např.: studium pozdních komorových potenciálů následujících za komplexem QRS, které mají malou amplitudu a jejich složky krátké trvání.

4. Využití záznamu a analýzy EKG v lékařské praxi

Metodika analýzy biosignálů je určena pro zpracování EKG vyplývající z potřeb podporovaných a spolupracujících lékařských pracovišť v oblastech:

- analýzy zátěžového a klidového EKG (ordinace preventivní kardiologie)
- podpory diagnostických metod analýzy a vizualizace multikanálového EKG - izopotenciálové mapy (II. interní klinika kardiologie a angiologie)
- zpracování Holterva monitorování EKG a záznamů EKG z implantabilních kardiostimulátorů (II. interní klinika kardiologie a angiologie)

V jednotlivých oblastech je dále přiblížen současný stav využití EKG a návazně navržena další možná aplikace analýzy biosignálů nebo směr podpory zpracování EKG s praktickým efektem v lékařské praxi.

4.1. EKG v oblasti preventivní kardiologie

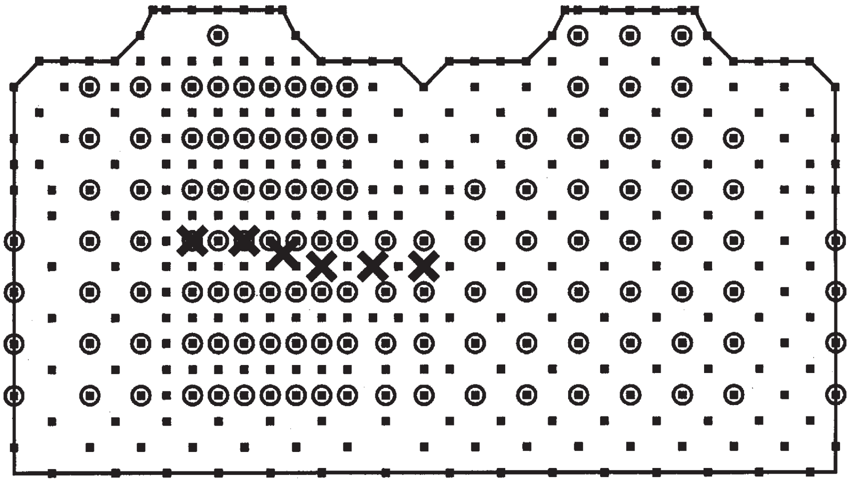
Na pracovišti kardiologická ambulance EuroMISE centra - Kardio je snímáno a počítačově ukládáno standardní 12-ti svodové EKG - klidové nebo zátěžové v kombinaci s ergometrem, u kterého lze volit programy zátěže. Komerčně dodávaný software nabízí základní vyhodnocení průměrů, průběhů srdeční frekvence a navržení jednoduché klasifikace typů křivek EKG. Systém podporuje databázové ukládání záznamů, export dat není jednoduše podporován.

U jednotlivých záznamů není dořešena filtrace rušivých vlivů okolí, či svalových potenciálů - izolinie je namodulována na nízkofrekvenčním signálu. V případě možnosti exportu celého navzorkovaného EKG lze rušivé vlivy odstranit pomocí filtrů ve frekvenční nebo časově-frekvenční oblasti.

Hlavní využití dalšího zpracování EKG je v oblasti hromadného vyhodnocení celé databáze záznamů.

4.2. Multikanálové EKG a izopotenciálové mapy

Na II. interní klinice kardiologie a angiologie Všeobecné fakultní nemocnice v Praze jsou diagnosticky využívány izopotenciálové mapy [6]. Mapování srdečních potenciálů vychází z multikanálového snímání EKG (příklad rozmístění elektrod 120-ti svodového EKG je na obr. 2)



Obrázek 2: Rozložení elektrod pro měření 120-ti kanálového EKG

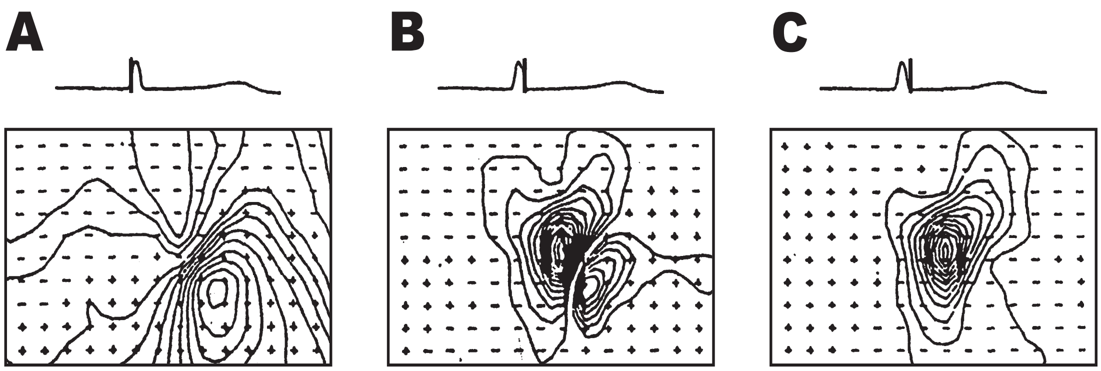
Na základě takto senímaných signálů EKG lze vytvořit izopotenciály, spojnice se stejnou úrovní elektropotenciálu. Z těchto izopotenciál pak vytvářet mapy (ukázka izopotenciálových map je na obr. 3).

V této oblasti je podpora směřována do unifikace dat snímaných multikanálovým EKG s různým počtem svodů. Výsledkem by měl být vstup do systému zobrazujícího izopotenciálové mapy, který je nezávislý na počtu svodů. Izopotenciálové mapy lze získat již z 12-ti svodového EKG, výsledeky zobrazení z menšího početu svodů jsou však značně nepřesné. V této oblasti lze použít metody aproximace a interpolace.

4.3. Záznam a analýza EKG u implantabilních kardiostimulátorů

Na II. interní klinice kardiologie a angiologie se zároveň využívá dlouhodobé Holtrovo monitorování EKG a také monitorování EKG v implantabilních kardiostimulátorech [7, 8].

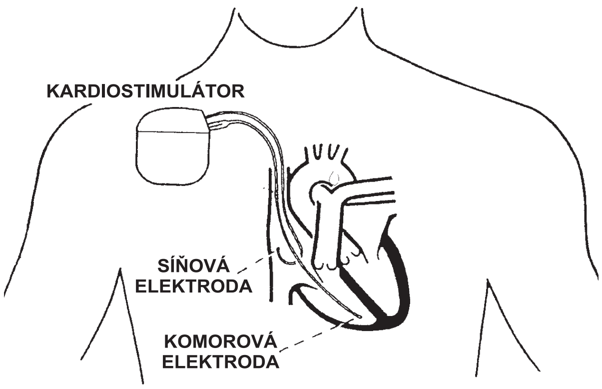
U ambulantního (Holtrova) monitorování EKG se aktivita srdce zaznamenává 24 hodin a posléze nahrává z přenosného přístroje a počítačově zpracovává. Není zaznamenáván celý signál, ale jen jeho významné



Obrázek 3: Ukázka izopotenciálových map

parametry, signál se však může nahrát pokud přístroj detekuje anomálii nebo po manuálním upozornění. Vyhodnocovány jsou především různé amplitudové histogramy v jednotlivých denních úsecích a frekvenční charakteristika celého průběh srdeční frekvence.

Využívané kardiostimulátory mají dvé bipolární elektrody, které slouží zároveň k stimulaci komory a síně a zároveň k intrakardiálnímu snímání EKG. Schematické znázornění používaného dvoudutinového kardiostimulátoru je na obr. 4.



Obrázek 4: Dvoudutinový kardiostimulátor

Tyto přístroje podporují oboustranný telemetrický přenos, který umožňuje nastavit a korigovat funkce kardiostimulátoru a zpracovávat záznamy EKG. I zde se zaznamenávají pouze parametry signálů. Okamžitě měřené hodnoty EKG z elektrod se však nevyužijí pouze v záznamech, ale především jako kontrolní a řídící vstup systému ovládajícího stimulační pulsy.

U softwaru dodávaného k Holtrovskému monitorování EKG i ve vyhodnocovací části programu určenému pro implantabilní kardiostimulátory není podporována analýza v časově-frekvenční oblasti. Tato analýza aplikovaná na části EKG i na průběh srdeční frekvence by měla zlepšit diagnostické vlastnosti záznamů.

5. Závěr

Z řady možností analýzy biosignálů zde byly přiblíženy metody frekvenční a především časově-frekvenční analýzy. Druhá metoda je vhodná pro různá vyhodnocení biosignálů srdce a zároveň není příliš rozšířena v lékařské praxi. Možnosti použití a aplikace byly ilustrovány na případech spolupracujících lékařských pracovišť.

Poděkování

Práce vznikla za podpory grantu MŠMT ČR LN00B107.

Literatura

[1] ***Svatoš, J.*: Biosignály z inženýrského pohledu (II. část)**; Lékař a technika, Praha: 1997: Vol: 28 No: 4 p. 80-87.

[2] ***Svatoš, J.*: Biologické signály I**; Vydavatelství ČVUT, Praha, 1998.

[3] ***Svatoš, J.*: Biosignály z inženýrského pohledu (III. část)**; Lékař a technika, Praha: 1997: Vol: 28 No: 6 p. 133-139.

[4] ***Hlaváč, V.; Sedláček, M.*: Zpracování signálů a obrazů**; Vydavatelství ČVUT, Praha, 2001.

[5] ***Jonáš, V.*: Klinická kardiologie**; Zdrav. naklad. Společnosti čs. lékařů, Praha, 1950, p. 558-686.

[6] ***Šťovíček, P. at al.*: QT Dispersion in 120 Electrocardiographic Leads in Patients with Structural Heart Disease**; Journal Paper, Pacing and clinical electrophysiology, 2002: Vol: 25 No: 1 p. 20-31.

[7] ***Němec, J. at al.*: Increase in heart rate precedes episodes of ventricular tachycardia and ventricular fibrillation in patients with implantable cardioverter defibrillators: analysis of spontaneous ventricular tachycardia database**; Journal Paper, Pacing and clinical electrophysiology, 1999: Vol: 22 No: 12 p. 1729-38.

[8] ***Novák, M.*: Holterovské funkce moderních implantabilních kardiostimulátorů**; Lékař a technika, Praha: 1998: Vol: 29 No: 2 p. 34-41.