Doktorandský den '98

Ústav informatiky
Akademie věd České republiky

Praha, 20. – 21. říjen 1998

Obsah

Doktorandský den ÚIVT '98 – program

— 20. říjen 1998 —

Časový rozvrh přednášek		
Dušan Bálek	Managing Component Interconnections in SOFA/DCUP Architectures	$8^{00} - 8^{35}$
Radovan Janeček	On Relations between the Glasgow Specification and the DCUP Component Model	$8^{35} - 9^{10}$
	přestávka 10 minut	
David Coufal	Fuzzy-PID hybridní regulátory	$9^{20} - 9^{55}$
Zdeněk Kestřánek	Numerické modelování některých případů kontaktního problému	$9^{55} - 10^{30}$
	přestávka 10 minut	
Marek Novický	Numerické řešení duální kontaktní úlohy v elasticitě	$10^{40} - 11^{15}$
Jan Klaschka	Aditivní rozklad funkce ve dvojím čase	$11^{15} - 11^{50}$

— 21. říjen 1998 —

Časový rozvrh přednášek		
Přemysl Žák	Buněčná síť– model neuronové sítě s proměnlivou topologií	$8^{00} - 8^{35}$
Arnošt Štědrý	Cascade Networks: Another Approach to Function Approximation	$8^{35} - 9^{10}$
	přestávka 10 minut	
Milan Rydvan	Biquadratic Error Functions for the BP-networks	$9^{20} - 9^{55}$
Petr Hanzlíček	Metody podpory rozhodování v medicíně	$9^{55} - 10^{30}$
	přestávka 10 minut	
Robert Mitka	Fuzzy logicke programovanie	$10^{40} - 11^{15}$

Managing Component Interconnections in SOFA/DCUP Architectures

doktorand:
MGR. DUŠAN BÁLEK
Malostranské nám. 25, Praha 1, 118 00
balek@nenya.ms.mff.cuni.cz

školitel:
DOC. ING. FRANTIŠEK PLÁŠIL, CSC.
Malostranské nám. 25, Praha 1, 118 00
plasil@nenya.ms.mff.cuni.cz

obor studia:
I2 - Softwarové systémy

Abstrakt

Nowadays, to allow for rapid software evolution, more and more software developers start to construct their products from reusable software components. The architecture of an application is described as a collection of components along with the interactions among these components. Even though the application's main building blocks are components, the properties of the application depend strongly on the character of the interconnections. This fact gave birth to a connector as an abstraction that provides software developers with a way to capture a character of an connection between components.

This paper deals with a problem of managing component interconnections in the SOFA/DCUP component model that has been developed at the Charles University and at the Academy of Sciences of the Czech Republic. One of the biggest issues of the DCUP architecture is to make applications composed from DCUP components dynamically updatable at runtime. The aim of this paper is to propose the solution that fully complies with this feature of our component model.

1. Introduction

Software developers frequently compose applications from a collection of interacting components. There are at least two strong arguments for employing component based style of programming:

- Around the world there is a number of software modules which offer services, therefore reusing them is desirable in order to facilitate the software development process [18].
- Programming using component technology is more effective for several reasons: it eliminates debugging of the reused parts, there are more opportunities for visual manipulation [17], and it makes it easier to arrange for a reconfiguration of an application [5, 19, 8].

1.1. Architecture Description Languages

There is a group of works [2, 3, 4, 5, 17, 13, 11] which focus on identifying proper abstractions for specifying an architecture of applications composed from software components. Essentially, all of them use some kind of configuration language (sometimes called ADL, Architecture Description Language), that allows interfaces of software components to be defined, and the structure of an application as a collection of mutually interconnected components to be specified.

1.2. SOFA Component Model Overview

In SOFA [19], an application is viewed as a hierarchy of software components. In analogy with the classical concepts of an object as an instance of a class, a *software component* is an instance of a *component template*. A "template" is referred to as a "component type". Basically a template is a framework which contains definitions of *implementation objects* and *nested components*. Every template is determined by its interface (set of services either *provided* or *required*), and by the definitions and *bindings* of implementation objects and nested components. The *SOFA CDL* (Component Definition Language) [22] is used to specify a component's interface and its architecture (in terms of nested components and their interconnections).

1.3. DCUP Architecture Overview

The DCUP architecture is a specific architecture of SOFA components which allows for their safe updating at runtime. It extends the SOFA component model (Section 1.2) in the following way:

- It introduces specific implementation objects (Component Managers, Component Builders, Updaters, ClassLoaders, and Wrappers).
- It makes the way components are interconnected more specific (every connection is mediated with a wrapper allowing a component to be locked during its update).
- It presents a technique for the updating of a component inside a running application.
- It specifies the necessary interaction between a running application and the rest of the world.

For more details on structure of DCUP components and dynamic updates we refer the reader to [19, 18].

Originally, the DCUP architecture was proposed to allow for updating applications running in the single address space. With moving towards distributed applications, the question of specifying character of component interconnections arises. Thus, the issue now is to propose a uniform way that allows for software developers to specify the quality (character) of a connection between components (e.g. local calls vs. remote calls).

2. Approaches to Capturing Component Interconnections

Studying related works, three different basic approaches to capturing component interconnections can be recognized. According to the terminology introduced in [15], these are:

- Implicit connections
- An enumerated set of built-in connectors
- User defined connecors

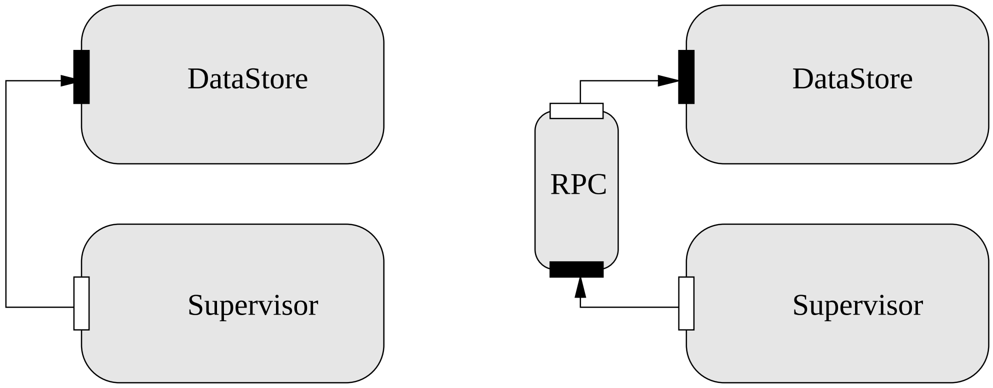
2.1. Implicit connections

The typical representative of this approach is the Darwin language [7]. The interconnections among components are specified in terms of binding `requires` to `provides` interfaces as ilustrated in the following piece of code.

```
instantiations
```

```
     DS: DataStore;
     S:  Supervisor;
bindings
     S.DataStoreAccess -- DS.DataStoreAccess;
```

The semantics of such connection is given by underlying environment (programming language, communication primitives of underlying operating system, etc.) and communicating components should be aware of it. Conformance checking of a connection can be based on `provides` to `requires` interface matching. The different semantics of the component interconnection has to be specified using special "communication" components (Figure 1).



Obrázek 1: Using "communication" components

2.2. An enumerated set of built-in connectors

The typical representative of this approach is the UniCon language [13, 14]. A user is provided with a choice to select from several predefined built-in connector types when specifying component interconnections. Usually, all the built-in connector types are intended to correspond to the usual communication primitives supported by underlying language or operating system. The semantics of a connection is simply given by the selected connector type. The connector type can be checked against the connected components' types to prove the conformance of the given interconnection. The following piece of code illustrates the idea of using built-in connectors.

```
implementation is
     uses DS interface DataStore
          end DS
     uses S interface Supervisor
          end S
     establish RemoteProcCall with
          S.DataStoreAccess as Caller
          DS.DataStoreAccess as Definer
          IDLTYPE(CORBA)
          end RemoteProcCall
end implementation
```

2.3. User defined connecors

The typical representative of this approach is the Wright language [3]. The connections among components are fully specified by users. The semantics of any language that belongs to this group should be rich enough for specifying the behavior of such user defined connectors. This behavior can be checked against the behavior of connected components to prove the conformance of such interconnection. For example, the above mentioned Wright language uses modified Hoare's CSP notation to specify sophisticated protocols of component interconnections.

3. Our Approach

To capture different types (semantics) of component interconnections, SOFA/DCUP uses the approach with an enumerated set of built-in connectors. To explain why, let us very briefly sketch the reasoning about the approaches described in the previous section:

- The first approach using implicit connectors together with special "communication" components does not distinguish between computations (represented by components) and connections between components.
- The approach using user defined connectors seems to be too universal with respect to automated code skeletons generation.
- There is only a (relatively) small collection of completely different communication primitives provided by the underlying environments and therefore the number of possible different communication semantics (types) is small too.
- Architecture description complexity increases by introducing the special communication components describing every connection between components.
- Architecture description complexity increases more by introducing user defined connector types to the language than it does by introducing built-in connector types.

In SOFA/DCUP, we distinguish three different possible roles that connectors can play in the overall architecture.

3.1. Connectors as Connection Semantics Descriptors

The primary role of connectors in the SOFA/DCUP architecture is to describe connection semantics. The aim is, based on the specified connector type, to automatically generate code (or code skeleton) for the implementation of the corresponding component interconnection.

According to the nature of connection semantics, connectors in the SOFA/DCUP architecture can be divided into two distinct classes:

- *Procedural connectors.* To this class belong e.g. local method calls, remote method calls using CORBA and/or Java RMI, and also event passing which is in fact invoking methods upon listener's interface. Component communication of this kind is typed. The type compatibility of `provides` and `requires` interfaces can be checked.
- *Data flow connectors.* To this class belong e.g. pipes (well known Unix's communication mechanism) or file accesses. Component communication of this kind is untyped. As a suitable abstraction, a stream of data can be used. There is no guarantee that a component reading data from a stream understands the data format.

Since our original motivation for introducing connectors to the SOFA/DCUP architecture was to move this architecture towards distributed applications, we will focus in the first step on building the procedural connectors into the existent SOFA/DCUP architecture. Later, we would like to also handle data flow connectors together with other possible connector classes.

3.2. Connectors as DCUP Wrappers

In the DCUP architecture, access to the services provided by a component is always mediated by a wrapper. Component wrappers implement the necessary locks needed to prevent anybody from outside of the component to access the component during its update. Since every connection between DCUP components has to be mediated by a wrapper, the possibility to implement wrapper functionality directly in the connector representing given connection naturally arises.

3.3. Connectors as Interface Adaptors

While building an application composed from reusable components, a system developer can realize the need for interconnecting two (or more) components with interfaces which do not exactly match. One possible solution to overcome this problem is to mediate such a connection with an adaptor converting calls between interfaces. At this moment several different levels of component interface incompatibility can be identified:

- Only minor differences breaking the underlying language's type compatibility rules with a clear correspondence between the methods of both interfaces; this allows for a fully automatic generation of the adaptor.
- Differences that allow to identify a correspondence between the methods of both interfaces in cooperation with a user, and, based on this information, to allow for semi-automatic generation of the adaptor.
- Differences that require the adaptors to be coded manually.

There are also other possible roles that adaptors can play (e.g. the role of a filter).

4. Conclusion

SOFA/DCUP is an ongoing research project. This paper presents our approach to the problem of managing component interconnections. In the SOFA/DCUP project we have decided to follow the approach used in the UniCon language with an enumerated set of built-in connectors (Section 2.2), although the possibility to use "communication" components (Section 2.1) in special cases is not excluded.

We have identified three potential roles that SOFA/DCUP connectors can play in the overall architecture: the role of connection semantics descriptors, of DCUP wrappers, and of interface adaptors, respectively.

As for the possible semantics of connections directly supported by SOFA/DCUP, in the beginning we plan to implement local method calls, remote method calls using CORBA and/or Java RMI, and event passing. In the future we plan to support also other semantics as well (e.g. pipes, file accesses).

References

[1] Peyman Oreizy: "Issues in the Runtime Modification of Software Architectures", *Tech. rep. UCI-ICS-TR-96-35*, University of California, Irvine, 1996

[2] D. Garlan, R. Monroe, D. Wile: "ACME: An Architecture Description Interchange Language". *Proceedings of CASCON '97*, 1997

[3] Robert J. Allen: "A Formal Approach to Software Architecture", *Ph.D. Thesis*, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1997

[4] R. Allen, D. Garlan: "Specifying Dynamism in Software Architectures", *In Proceedings of the Foundations of Component-Based Systems Workshop*, 1997

[5] K. M. Goudarzi, J. Kramer: "Maintaining Node Consistency in the Face of Dynamic Change", *Proceedings of the Third International Conference on Configurable Distributed Systems*, 1996

[6] J. Kramer, J. Magee: "Analysing Dynamic Change in Software Architectures: A Case Study", *Proceedings of the Fourth International Conference on Configurable Distributed Systems*, 1998

[7] J. Magee, N. Dulay, J. Kramer: "Regis: A ConstructiveDevelopment Environment for Distributed Programs", *Distributed Systems Engineering Journal*, 1(5), 1994

[8] L. Bellissard, S. B. Atallah, F. Boyer, M. Riveill: "Distributed Application Configuration", *Proceedings of the Sixteenth International Conference on Distributed Computing Systems*, 1996

[9] Ch. Hofmeister: "Dynamic Reconfiguration of Distributed Applications", *Ph.D. Thesis*, University of Maryland, 1993

[10] T. Mittermeir, L. Wurfl: "Composing Software from Partially Fitting Components", *Proceedings of IPMU'96*, 1996

[11] D. C. Luckham, J. J. Kenney, L. M. Augustin, J. Vera, D.Bryan, W. Mann: "Specification and Analysis of System Architecture Using Rapide", *IEEE Transactions on Software Engineering*, 21(4), 1995

[12] J. M. Purtilo: "The Polylith Software Bus", *ACM Transactions on Programming Languages and Systems*, 16(1), 1994

[13] M. Shaw, R. DeLine, D. V. Klein, T. L. Ross, D. M. Young, G. Zalesnik: "Abstractions for Software Architecture and Tools to Support Them", *IEEE Transactions on Software Engineering*, 21(4), 1995

[14] M. Shaw, R. DeLine, G. Zalesnik: "Abstractions and Implementations for Architectural Connections", *Proceedings of the Third International Conference on Configurable Distributed Systems*, 1996

[15] J. Bishop, R. Faria: "Connectors in Configuration Programming Languages: are They Necessary?" *Proceedings of the Third International Conference on Configurable Distributed Systems*, 1996

[16] Neno Medvidovic: "A Classification and Comparison Framework for Software Architecture Description Languages", *Tech. Rep. UCI-ICS-TR-97-02*, University of California, Irvine, 1997

[17] "JavaBeans 1.0 Specification", http://splash.javasoft.com/beans/spec.html

[18] W. Pree: "Framework Patterns". *SIGS Books & Multimedia*, 1996

[19] V. Issarny, Ch. Bidan: "Aster: A Framework for SoundCustomization of Distributed Runtime Systems". *Proceedings of the Sixteenth International Conference on Distributed Computing Systems*, 1996

[20] F. Plasil, D. Balek, R. Janecek: "DCUP: Dynamic Component Updating in Java/CORBA Environment", *Tech. Report No. 97/10, Dep. of SW Engineering*, Charles University, Prague.

[21] F. Plasil, D. Balek, R. Janecek: "SOFA/DCUP:Architecture for Component Trading and Dynamic Updating", *Proceedings of the Fourth International Conference on Configurable Distributed Systems*, Annapolis, Maryland, USA, IEEE CS Press 1998.

[22] V. Mencl: "Component Definition Language", *Master Thesis, Dep. of SW Engineering*, Charles University, Prague, 1998.

On Relations between the Glasgow Specification and the DCUP Component Model

doktorand:
MGR. RADOVAN JANEČEK
Malostranské nám. 25, Praha 1, 110 00
janecek@nenya.ms.mff.cuni.cz

školitel:
DOC. ING. FRANTIŠEK PLÁŠIL, CSC.
Malostranské nám. 25, Praha 1, 110 00
plasil@nenya.ms.mff.cuni.cz

obor studia:
I2 - Softwarové systémy

Abstrakt

Current experiences of software developers and system architectures designers prove an idea stemming from scientific communities. The idea that the traditional concept of software components does not provide sufficient expressive power for describing complex software systems. The concept introduces components as an atomic (or indivisible) entities the software systems are built from. Therefore, recently emerged component models allow designers to create component hierarchies by nesting components into another ones. This paper compares the DCUP component model evolved at the Charles University and at the Academy of Sciences of the Czech Republic with the JavaSoft's proposal of an Extensible Runtime Containment and Services Protocol for JavaBeans. It shows similarities and differences between these two approaches. Furthermore, it points out weaknesses of the JavaSoft's model, proposes improvements of that model and imposes certain rules of its use to provide the same functionality as the DCUP one.

1. Introduction

Nowadays, development process of software systems got so complex and time consuming that a strong need for rapid and ease software reuse emerged. Thus, many designers shifted their attention from object-oriented programming to a so called component-based one that offers more coarse granularity of software reuse. The author of this paper took part in the development of the specific component model (DCUP) that allows dynamic updating of components even at runtime of applications composed from these DCUP components. This model also introduces concept of component hierarchy that will be explained later.

Being aware of the fact that it is not easy to live in this world and not to comply with standards at the same time, the paper tries to exploit widely used standard called JavaBeans and Extensible Runtime Containment and Services Protocol [15] to achieve the functionality of the DCUP component model.

1.1. Component Hierarchies

In addition to better software reuse, there is another justification of using software components: an easier and more rapid building of component based applications. Unfortunately, from this point of view, many current component models including JavaBeans have not satisfied expectation of many software developers because there is no way how to reuse more complex software units composed from several mutually interconnected components. These models consider only a single component as a framework of objects that can be manipulated or customized. Basically, the notion of component hierarchies raises granularity of components to even more coarse and general level because it allows to perform operations on component frameworks. This approach allows to design the architecture of an application by means of interconnections and customizations of these frameworks.

1.2. Architecture Definition Languages

There is a group of works dealing with architecture definition languages [2, 3, 4, 5, 17]. These meta level languages offer means for defining the structure of an application as a set of components and/or component frameworks that are mutually interconnected (possibly via special entities called connectors).

1.3. DCUP Overview

The DCUP component model is an extension of the SOFA component model [19]. In SOFA, an application is viewed as a hierarchy of software components. This hierarchy is described by Component Definition Language (CDL) that is used for defining component `templates`. In analogy with the classical concepts of an object as an instance of a class, a component is an instance of a component template. From an architectural point of view, the components described by that language can be divided into two classes: *primitive* components that don't have any nested components and *composite* components nesting other composite and/or primitive ones [6]. A particular component description specifies component's interfaces that the component `provides` and/or `requires` and component's `architecture` that contains a description of an internal framework of nested components and/or implementation object instances. It implies that component's interfaces can be implemented either by internal implementation objects or by `provides` interfaces of nested components. A component cannot provide any services until its required services are supplied to it from the outside world (nesting component and/or other sibling components).

1.4. Glasgow Overview

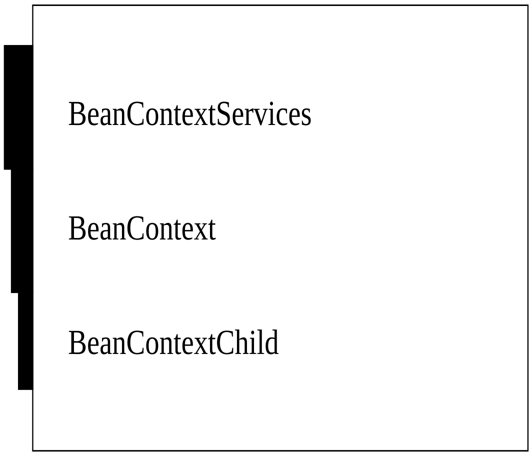
Glasgow [16] is the code-name for the new specification of the JavaBeans component model. Glasgow provides standard means for creating more sophisticated JavaBeans components and their integration with runtime environment (i.e., desktop, web browser, etc.). To achieve this, Glasgow adds three new capabilities to the JavaBeans component model:

- The Extensible Runtime Containment and Services Protocol (v0.99, Aug.24)
- The Drag and Drop Subsystem for the Java Foundation Classes (v0.96, Aug.24)
- The JavaBeans Activation Framework (Released, Mar.17)

It is the first capability that is of interest of this paper. It defines a set of interfaces that supports extensible mechanisms that:

- Introduce an abstraction for the environment, or context, in which JavaBeans logically function. These contexts create a hierarchy or structure of JavaBeans.
- Enable the dynamic addition of arbitrary services to a JavaBean's context.
- Provide a mechanism for discovering services available within the context.

- Provide a simple mechanism to propagate an Environment to a JavaBean.
- Support JavaBeans that are also Applets.



Obrázek 2: Usual implementation of an environment

A comprehensive description of these interfaces goes out of scope of this paper. Briefly, there is an interface `BeanContextServices` that extends `BeanContext` that extends `BeanContextChild` interface. `BeanContextChild` interface provides a JavaBean with knowledge of its nesting context. `BeanContext` provides its children with access to its resources and exports add/remove operations. Finally, `BeanContextServices` defines methods for discovering and using services registered with it. These services are registered as Java classes along with a reference to an object implementing `BeanContextServiceProvider` interface. Thus, that object can be viewed as a factory of registered service. Obviously, these services are not intended to be used by any out-of-context entities but children of a particular `BeanContext`.

For brevity, the name Glasgow will denote the first part of the Glasgow specification - the Extensible Runtime Containment and Services Protocol.

2. DCUP and Glasgow Compared

2.1. How They Fit Together

At first, both models introduce a notion of a component hierarchy. An application structure follows a logical (tree-like) hierarchy of components. This hierarchy may change during the lifetime of an application in both cases. It means that any component can be removed from or added to the hierarchy at runtime. The important thing is that, in contrast to a single component, the hierarchy can be manipulated as a whole. At second, all DCUP nested components (resp. `BeanContextChilds`) have access to certain set of services that are available within nesting DCUP component (resp. `BeanContext`). At third, it is the nesting entity that is responsible for proper externalization of an internal state of all nested entities (components or objects).

2.2. How They Differ from Each Other

Although both models provide a reasonable set of means for creating and manipulating component hierarchies, there are several differences between them. This section tries to point out the most important ones.

2.2.1 Motivation: Probably, this difference is the most substantial one because it is a common cause of the others. The core of both models' motivations is the same: to provide a set of means for logical grouping of software components and handling these groups in the same manner as single components are handled.

However, the purpose of grouping DCUP components is to create more complex component. This "super-component" provides a set of services that are implemented by nested components or by the super-component itself. Thus, nested components can be regarded as implementation details of a super-component. Main motivation of the DCUP component model is to provide small set of abstractions supporting dynamic changes of an application even at runtime. Apparently, dividing an application into several pieces that follows component hierarchy allows performing these changes on affected parts only without stopping the rest of the application.

On the other hand, the Glasgow's purpose of grouping JavaBeans into a specific context is to provide a common living place for them. JavaBeans grouped together can share the same resources and utilize the same set of services. Moreover, they can interrogate its context to get knowledge of other sibling JavaBeans.

2.2.2 Granularity of Entities Residing in the Hierarchies: While any `BeanContext` is able to instantiate any Java classes, a DCUP component can contain only DCUP components as a child components. Other parts of a DCUP component are anonymous internal objects.

2.2.3 Inter-component Contracts Specification: Main difference between the discussed models is that all DCUP components have defined two sets of interfaces specifying their provisions and requirements while the Glasgow model relies on standard JavaBeans concept of interfaces. This concept is able to express component provisions only. Moreover, there is no rule saying that a `BeanContext` has to implement any other interfaces except the `BeanContext` interface.

2.2.4 Dynamic Changes in the Hierarchies: Both models allow performing dynamic changes. The Glasgow one is more general in this case but does not ensure the integrity of an affected system. Dynamic addition and/or removal can be initiated by third party and vetoed by any outside entity. The DCUP approach ensures safe shift of target part of component hierarchy to a new version. The state of the former version is transformed to a new one.

3. Dynamic Glasgow along the DCUP Lines

Previous section showed that the Glasgow component model has few weak points with respect to means offered for architecture design and maintenance. Fortunately, the specification does not impose strong rules of its use. Thus, it allows defining own (DCUP) semantics of particular methods without violating the standards. This section proposes steps necessary in order to create Glasgow hierarchy keeping the functionality of the DCUP model. To be able to understand this section readers should read [19] first.

The first task is to map all important abstractions of DCUP to Glasgow. The permanent part of a DCUP component should be represented by the object implementing `BeanContextChild` interface. It implies that the object will play the role of `CManager`. The heart of the replaceable part of a component, `CBuilder`, can be implemented as an ordinary implementation object inside a context. It is a `CBuilder` responsibility to add/remove all child entities to/from its `BeanContext`. To be nested in a "super-component", all DCUP components should implement at least `BeanContextChild` but in most cases they will implement `BeanContextServices`.

As mentioned in sections 2.2.1 and 2.2.3, the Glasgow components can utilize services available within a `BeanContext` but these services are provided indirectly by `BeanContextServiceProvider` not by the context itself. To achieve a DCUP functionality it is necessary to impose strong rules on these providers. At first, the object implementing this interface should be a DCUP component resp. it should implement `BeanContextChild` interface at least. It also implies that such a provider cannot be registered with more than one context. Because DCUP components can use particular instances of a service, it is desirable to exploit `ServiceSelector` object in order to select a required instance. Furthermore, these providers should be divided into two classes: providers that provide services to the outside world and 'internal services providers'. The latter providers can follow the Glasgow semantics. However, the providers exporting services outside of their context have to keep the immutable set of services defined in `provides` part of CDL description of a component. Supplying services required by a child should be done via firing `serviceAvailableEvent`.

Inspired by Glasgow, the DCUP model should adopt flexibility of dynamic addition and removal of children within a particular `BeanContext`. It will improve dynamic reconfiguration capabilities of DCUP.

4. Conclusion

This paper briefly compared two component models that allow creating component hierarchies (Glasgow and DCUP). It pointed out weak properties of the Glasgow model and proposed a way how to map DCUP abstractions into Glasgow terms. A feature of this mapping is that it does not violate any Glasgow rules and provides all advanced properties of the DCUP model such as dynamic updating of components even at runtime of affected applications.

References

[1] Peyman Oreizy: "Issues in the Runtime Modification of Software Architectures", *Tech. rep. UCI-ICS-TR-96-35*, University of California, Irvine, 1996

[2] D. Garlan, R. Monroe, D. Wile: "ACME: An Architecture Description Interchange Language". *Proceedings of CASCON '97*, 1997

[3] Robert J. Allen: " A Formal Approach to Software Architecture", *Ph.D. Thesis*, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1997

[4] R. Allen, D. Garlan: "Specifying Dynamism in Software Architectures", *In Proceedings of the Foundations of Component-Based Systems Workshop*, 1997

[5] K. M. Goudarzi, J. Kramer: "Maintaining Node Consistency in the Face of Dynamic Change", *Proceedings of the Third International Conference on Configurable Distributed Systems*, 1996

[6] J. Kramer, J. Magee: "Analyzing Dynamic Change in Software Architectures: A Case Study", *Proceedings of the Fourth International Conference on Configurable Distributed Systems*, 1998

[7] G. Abowd, R. Allen: "Using Style to Understand Descriptions of Software Architecture" *ACM SIGSOFT'93 Symposium on Foundations of SW Engineering*, 1993

[8] R. Allen, D. Garlan: "Formalizing Architectural Connection" *Proceedings of the Sixteenth International Conference on SW Engineering*, 1994

[9] L. Deri: Droplets: "Breaking Monolithic Applications Apart", IBM Research Division, Zurich, April 1996

[10] D. Garlan, R. Allen, J. Ockerbloom: "Exploiting Style in Architectural Design Environments", *ACM SIGSOFT'94 Symposium on Foundations of SW Engineering*, 1994

[11] Ch. Hofmeister: "Dynamic Reconfiguration of Distributed Applications", Ph.D. Thesis, University of Maryland, 1993

[12] T. Mittermeir, L. Wurfl: "Composing Software from Partially Fitting Components", *Proceedings of IPMU'96*, 1996

[13] JavaSoft: "JavaBeans 1.0 API Specification", *http://splash.javasoft.com/beans/spec.html*

[14] D. C. Luckham, J. J. Kenney, L. M. Augustin, J. Vera, D.Bryan, W. Mann: "Specification and Analysis of System Architecture Using Rapide", *IEEE Transactions on Software Engineering*, 21(4), 1995

[15] JavaSoft: "Extensible Runtime Containment and Services Protocol for JavaBeans", *http://java.sun.com/beans/glasgow/context99.ps*

[16] JavaSoft: "Glasgow Specification", *http://java.sun.com/beans/glasgow/*

[17] Neno Medvidovic: "A Classification and Comparison Framework for Software Architecture Description Languages", *Tech. Rep. UCI-ICS-TR-97-02*, University of California, Irvine, 1997

[18] F. Plasil, D. Balek, R. Janecek: "DCUP: Dynamic Component Updating in Java/CORBA Environment", *Tech. Report No. 97/10, Dep. of SW Engineering*, Charles University, Prague.

[19] F. Plasil, D. Balek, R. Janecek: " SOFA/DCUP:Architecture for Component Trading and Dynamic Updating", *Proceedings of ICCDS'98*, Annapolis, Maryland, USA, IEEE CS Press 1998.

Fuzzy-PID hybridní regulátory

doktorand:
ING. DAVID COUFAL
Ústav informatiky AV ČR, Pod Vodárenskou věží 2
182 07 Praha 8
coufal@uivt.cas.cz

školitel:
DOC. ING. STANISLAV KREJČÍ, CSC.
KŘPVT, Univerzita Pardubice, Nám. Čs. Legií 565
532 10 Pardubice
krejci@upce.cz

obor studia:
technická kybernetika

Abstrakt

Příspěvek předkládá výsledek rešeršní analýzy problematiky fuzzy-PID hybridních regulátorů. Fuzzy-PID hybridní regulátory se jeví jako efektivní nástroje pro regulaci složitých soustav, zachovávající si výhodné vlastnosti konvenčních PID regulátorů a zároveň využívající "model-free" povahy fuzzy regulátorů. V současné literatuře se objevuje několik přístupů k návrhu těchto hybridních struktur. Příspěvek referuje jednotlivé přístupy a pokouší se je určitým způsobem systematizovat.

1. Důležité pojmy a zkratky

- FIB - fuzzy inferenční báze. FIB je primárně charakterizována množinou vstupních a množinou výstupních veličin. Sekundárně je charakterizována počtem a typem použitých fuzzy pravidel a druhem použité fuzzy inference. FIB na základě své vnitřní struktury realizuje zobrazení z množiny vstupních veličin do množiny výstupních veličin.
- FŘS - fuzzy řídící struktura. Tímto pojmem je označován člen regulačního obvodu ovlivňující činnost PID regulátoru. FŘS je vnitřně tvořena jednou nebo více FIB a dodatečnými výpočetními algoritmy. FŘS ve spojení s PID regulátorem představuje fuzzy-PID hybridní regulátor.
- K_p, K_i, K_d, T_i, T_d označují po řadě proporcionální, integrační a derivační zesílení, integrační a derivační časovou konstantu PID regulátoru.

2. Úvod

V současné době jsou PID regulátory stále nejrozšířenějším druhem regulátorů používaných v technické praxi. Tato skutečnost vychází z jejich relativní jednoduchosti, snadnosti hardwarové implementace a možnosti využití aparátu hluboce propracované teorie PID regulace. Z této teorie plyne, že PID regulátor s konstantně nastavenými parametry je zcela postačující pro regulaci lineárních procesů prvního a druhého řádu. V reálných podmínkách je však linearita spíše vyjímkou než pravidlem a často se setkáváme s takovými jevy jako jsou hystereze, dopravní zpoždění, saturace atd. Obecně lze říci, že popis procesu pomocí jednoduchého matematického modelu je u většiny reálných regulovaných soustav obtížný nebo zcela selhává. Příklad tohoto faktu je v [3] výstižně vyjádřen větou "The one thing you can say about time constant is that it is rarely constant". Použití PID regulátorů s konstantně nastavenými parametry má pak v případě nelineárních procesů často za následek nízkou efektivitu regulačního pochodu.

Z tohoto důvodu vzniká požadavek na vývoj vhodnějších algoritmů implementovaných v nových typech regulátorů. Jednou z možných cest je adaptivní regulace. Adaptivní regulace je dnes široce rozvinutá část teorie řízení, která byla a je úspěšně aplikována v některých případech neřešitelných pomocí klasické regulace. Její slabou stránkou je ovšem nutnost práce s apriorní znalostí matematického modelu regulovaného procesu, což je v některých případech poměrně silný požadavek. Tuto apriorní znalost nevyžadují regulační algoritmy založené na tzv. soft-computing výpočetních metodách, jako jsou fuzzy inferenční techniky, neuronové sítě a genetické algoritmy. V oblasti regulace procesů má zvláště použití fuzzy inferenčních technik dlouhou tradici a jeho úspěšnost při regulaci těžko modelovatelných procesů bývá prezentována na několika notoricky známých příkladech.

Jedním z kritických míst regulátorů konstruovaných na těchto principech je otázka stability, přestože je v této oblasti prováděn intenzivní výzkum, je její řešení složitější než u PID regulátorů. Použití soft-computing výpočetních metod tedy vnáší nový pohled do návrhu regulačních algoritmů, ale nese s sebou určité nedostatky nevyskytující se u klasické PID regulace. Proto se jako zajímavá jeví kombinace konvenčního PID regulátoru s řídící strukturou implementovanou na některém z výše uvedených principů.

Vzhledem k tomu, že použití fuzzy inferenčních technik je v oblasti řízení procesů nejrozšířenější zabývá se tento příspěvek pouze hybridními strukturami typu fuzzy-PID. Jádrem takovýchto struktur je konvenční PID regulátor jehož činnost je ovlivňována vyšší fuzzy řídící strukturou - FŘS. Cílem hybridní struktury je zlepšení regulačního pochodu oproti použití samostatného PID regulátoru, při zachování některých užitečných vlastností vykazovaných PID regulací, např. již výše zmíněná stabilita. Hybridní struktury obecně jsou vhodné i z důvodu ekonomických. Upgrade stávajících PID instalací se provádí pouze dodatečným přidáním mikročipu s implementovaným řídícím algoritmem, tedy nedochází k výměně stávajících regulátorů, ale pouze k jejich rozšíření.

Je přirozené, že různí autoři volí různé přístupy k návrhu fuzzy-PID hybridních struktur, přesto ale, lze na základě analýzy architektury propojení procesu s regulátorem a FŘS provést základní kategorizaci. Následné další třídění vychází z typu předávaných informací či jiných podrobnějších znaků. Na základě této strukturně informační analýzy je možné provést rozdělení stávajících přístupů následovně:

- I. fuzzy supervising

    kategorie IA - lokalní popis

    kategorie IB - globální popis

    kategorie IC - parametrizovaná Ziegler-Nichols formule

- II. fuzzy precompensator

V případě kategorie I, FŘS vystupuje jako supervisor a na základě svých vstupů nesoucích informace o stavu regulovaného procesu nastavuje parametry PID regulátoru. V případě kategorie II, vychází regulační schéma z myšlenky dynamické úpravy žádané hodnoty během regulačního pochodu fuzzy řídící strukturou, úprava je prováděna tak, aby PID regulátor vycházející z této hodnoty lépe reguloval daný proces. V tomto případě FŘS vystupuje jako tzv. precompensator.

Rozdělení první kategorie do tří podkategorií reflektuje charakter informací vstupujících do/vystupujících z FŘS.

Pro kategorii IA jsou vstupy fuzzy supervisora veličiny popisující aktuální stav regulačního pochodu, jako jsou regulační odchylka a její inkrement. Výstupy jsou hodnoty parametrů PID regulátoru.

Kategorie IB zahrnuje případy kdy vstupem fuzzy supervisora jsou hodnoty veličin v jistém smyslu charakterizující kvalitu probíhajícího regulačního pochodu. Typickými veličinami tohoto druhu jsou např. přeregulování, doba náběhu atd. Na základě těchto veličin pak supervisor nastavuje aktuální hodnoty parametrů regulátoru.

Kategorie IC je charakterizována v případě vstupů stejně jako kategorie IA, výstupem fuzzy supervizora je ale hodnota pomocného(ných) parametru(ů), které jsou dále použity k nastavení parametrů PID regulátoru pomocí parametrizované Ziegler-Nichols formule (ZN formule)

V následujících kapitolách jsou probrány jednotlivé kategorie s příslušnými odkazy na literaturu.

3. Kategorie IA

Tuto kategorii reprezentují články [1, 2, 3]. Navrhované fuzzy řídící struktury jsou charakterizovány vstupními veličinami popisujícími aktuální stav regulačního pochodu - regulační odchylkou, inkrementem regulační odchylky za periodu vzorkování resp. aktuální hodnotou regulované veličiny. Výstupy jsou přímo parametry PID regulátoru, eventuálně veličiny z nichž lze tyto parametry získat na základě jednoduchých vztahů.

Autoři článku [1] uvažují diskrétní PID regulátor ve tvaru:

$$u(k) = K_p e(k) + K_i T_s \sum_{i=1}^{k} e(i) + \frac{K_d}{T_s} \Delta e(k),$$

kde $e(k)$ značí regulační odchylku, $\Delta e(k) = e(k) - e(k-1)$ její inkrement, T_s interval vzorkování a $u(k)$ výstup regulátoru. Parametry regulátoru K_p, K_i a K_d jsou předmětem on-line nastavování na základě výstupů FŘS. Důležitým předpokladem, z něhož autoři při návrhu vycházejí je znalost rozmezí v kterých se mohou pohybovat hodnoty zesílení K_p a K_d během regulačního pochodu. Tato rozmezí jsou určena intervaly $[K_{p,min}, K_{p,max}]$ a $[K_{d,min}, K_{d,max}]$. Pro případ, že nelze tyto meze určit apriori, navrhují autoři heuristická pravidla pro jejich určení z kritického zesílení K_u a kritické periody T_u. Význam K_u a T_u je stejný jako při použití ZN formule. Uvedený předpoklad umožňuje normalizovat K_p a K_d pomocí vztahů:

$$\begin{aligned} K_p^{'} &= (K_p - K_{p,min})/(K_{p,max} - K_{p,min}), \\ K_d^{'} &= (K_d - K_{d,min})/(K_{d,max} - K_{d,min}). \end{aligned}$$

Hodnoty $K_p^{'}$ a $K_d^{'}$ pak leží v intervalu $[0, 1]$. Integrální zesílení K_i je vázáno s proporcionálním a derivačním vztahem $K_i = K_p^2/(\alpha K_d)$, kde α je parametr. Tato rovnice, uvažujeme-li v časových konstantách, vyjadřuje lineární závislost T_i na T_d: $T_i = \alpha T_d$.

Fuzzy řídící struktura je vnitřně tvořena třemi FIB. Každá určuje hodnotu jednoho z parametrů $K_p^{'}, K_d^{'}$ a α. Pro všechny tři FIB je vstupem regulační odchylka $e(k)$ a její inkrement $\Delta e(k)$.

V článku je diskutována otázka stability regulačního obvodu s tímto druhem regulace. Protože přímá matematická analýza tohoto problému je obtížná, autoři navrhují přidat do regulační smyčky další člen, který monitoruje možné náznaky nestabilního chování na základě vhodných kritérií. Pokud je zjištěno nestabilní chování je provedena adekvátní korekční akce, např. parametry PID regulátoru jsou nastaveny na hodnoty, o nichž se ví, že mají stabilizující účinek. Jedním z kritérií dle něhož je možné indikovat nestabilitu je monitorování magnitudy píků. Systém je považován za nestabilní pokud se magnituda píků během regulačního pochodu třikrát za sebou zvětší.

Výchozím bodem autorů konferenčního příspěvku [2] je konvenční PID gain scheduling. Při tomto typu regulace je prostor stavů nelineárního procesu rozdělen na několik oblastí. Pro každou oblast se předpokládá lineární chování a jsou zvoleny konstantní hodnoty paramerů PID regulátoru pro adekvátní regulaci procesu v dané oblasti. V průběhu regulačního pochodu jsou pak parametry regulátoru nastavovány na základě toho, v jaké oblasti se regulovaný proces právě nachází. Nevýhodou této techniky jsou skokové změny parametrů při přechodu z jedné oblasti do druhé. Proto autoři navrhují rozšířit tento koncept o vyšší FŘS, která umožňuje interpolaci hodnot parametrů při pohybu v prostoru stavů. Tato interpolace je prováděna na základě lokálních nastavení parametrů pro jednotlivé oblasti a umožňuje hladký průběh jejich změn při přechodu z jedné oblasti do druhé.

Navrhovaná FŘS je tvořena jednou FIB. Vstupem FIB je aktuální hodnota regulované veličiny. FIB je tvořena pravidly Takagi-Sugenovského typu. Jejich počet je dán počtem oblastí pro které jsou zvoleny lokální nastavení parametrů regulátoru. Důležitým poznatkem je, že v konsekventu pravidel vystupují reciproké hodnoty jednotlivých zesílení PID regulátoru a nikoliv jejich přímé hodnoty. Tento fakt je v článku teoreticky zdůvodněn na základě rozboru ITAE pravidel pro nastavování parametrů PID regulátoru. Na simulacích provedených v závěru článku je ukázáno, že výsledná efektivita regulace je vyšší, než kdyby se v konsekventu používali přímo hodnoty jednotlivých zesílení.

Na článek [2] těsně navazuje článek [3], popisující návrh komplexního gain scheduleru, využívající algoritmus navržený v [2].

4. Kategorie IB

Přístup zvolený v článcích [4, 5, 6, 7] uvažuje FŘS se vstupními veličinami popisujícími kvalitu regulačního pochodu. Těmito veličinami jsou např. *přeregulování, perioda oscilace, rychlost útlumu* atd. Narozdíl od případu, kdy vstupními veličinami FŘS jsou regulační odchylka a její inkrement, lze předpokládat, že na základě těchto veličin nesoucích globálnější informaci je možné nastavit parametry regulátoru vhodněji. Tento klad je ovšem vyvážen větší výpočetní náročností než v případě kategorie IA a nutností přidat do regulačního obvodu další člen, který monitoruje hodnoty těchto kvalitativních indikátorů.

Článek [4] se zabývá návrhem fuzzy supervisora pro regulátor udržující na předepsané úrovni výšku hladiny v kotli parního generátoru atomového reaktoru. Použitý regulátor je specifického tvaru, ale po určitém přiblížení ho lze chápat jako PI regulátor. Předmětem supervisingu jsou parametry K_p a K_i. Fuzzy supervisor nastavuje jednotlivé parametry na základě čtyř kvalitativních indikátorů, představujících jeho vstupní veličiny. Těmito veličinami jsou *pseudo-útlumová rychlost, oscilační stupeň, ofset* a *přeregulování*.

- *pseudo-útlumová rychlost PDR* je dána poměrem:

  $$PDR = \frac{r(t_2)}{r(t_1)}, \text{ kde } r(t) = (e(t) + p(e'(t))^2.$$

  $e(t)$ je regulační odchylka v čase t, $p > 0$ je váha, t_1 a t_2 jsou dva po sobě jdoucí vzorkovací okamžiky ($t_1 < t_2$). Vzhledem k tomu, že $r(t)$ je funkcí regulační odchylky a její derivace, popisuje $r(t)$ jak okamžitý stav regulačního pochodu tak i jeho blízkou budoucnost. Malé hodnoty $r(t)$ znamenají, že $e(t)$ i její derivace $e'(t)$ mají malé absolutní hodnoty, a tedy, že regulovaný proces je v blízkosti požadovaného stavu a v její blízké budoucnosti nedojde k nějakým významným změnám. Naopak velké hodnoty $r(t)$ signalizují buď velkou odchylku od žádaného stavu nebo významné změny v blízké budoucnosti. Pokud $PDR < 1$ pro všechna t, tedy $r(t)$ monotónně klesá, je to známkou toho, že regulace probíhá žádoucím způsobem. V případě opačném $PDR > 1$ pro všechna t, jeví proces nestabilní chování a je třeba změnit nastavení parametrů regulátoru.

- definice *oscilačního stupně* vychází z trajektorie regulačního pochodu v rovině souřadnic $p^{1/2}e'(t), e(t)$. Blíže viz obrázek v originálním článku.

- hodnota *ofsetu* OF je definována rovnicí:

$$OF = \sum_{i=n-k}^{n} e(t_i),$$

i je index okamžiku vzorkování, k je délka časového pásma v intervalech vzorkování, za které je hodnota ofsetu počítána, t_n označuje současný časový okamžik. Ofset zhruba charakterizuje zda-li je hodnota regulované veličiny v blízkosti žádané hodnoty.

- *přeregulování* je dáno největší absolutní odchylkou regulované veličiny od žádané hodnoty v rámci jedné periody pozorování.

Vnitřně je supervisor tvořen dvěmi dvojicemi FIB, jedna dvojice řídí nastavování hodnoty K_p, a druhá nastavování hodnoty K_i.

Poněkud odlišný od ostatních článků této kategorie je článek [5]. Článek se zabývá návrhem expertního systému pro nastavování parametrů PID regulátoru - K_p, T_i a T_d. Znalostní báze expertního systému je obdobně jako u FŘS reprezentována souborem pravidel. V případě tohoto expertního systému se však v pravidlech nepoužívají lingvistické termy označující fuzzy množiny a následně také inferenční proces nevychází z konceptu fuzzy inferenčních technik. Přesto, ale je článek přínosný k problematice fuzzy-PID hybridních struktur tím, že jak i autor sám v závěru podotýká, rozšíření navrženého expertního systému o fuzzy inferenční techniky je velmi přirozené. Z hlediska přímého použití při návrhu FŘS je podnětný použitý způsob získání pravidel znalostní báze expertního systému. Expertní systém primárně pracuje s pěti kvalitativními indikátory, jako vstupními veličinami. Těmito veličinami jsou 1) *přeregulování*, 2) *poměr druhého píku přeregulování k prvnímu*, 3) *perioda oscilace*, 4) *doba náběhu* a 5) *útlum*. Cílem expertního systému je nastavit parametry regulátoru tak, aby regulační pochod vykazoval požadovaný průběh. Pravidla znalostní báze mají tvar:

```
zvýšení Kp snižuje periodu oscilace a naopak
zvýšení Kp zvyšuje přeregulování a naopak
snížení Ti snižuje poměr přeregulování a naopak
zvýšení Ti snižuje přeregulování a naopak
zvýšení Td snižuje dobu náběhu a naopak
atd ...
```

Pravidla jsou získána rozborem tzv. přehledové mapy (tuning map) systému druhého řádu s dopravním zpožděním. Přehledová mapa nějakého systému je soubor graficky znázorněných průběhů regulačních pochodů při regulaci tohoto systému PID regulátorem pro rozsáhlou množinu různých nastavení parametrů regulátoru. Přehledová mapa tak v grafické podobě představuje souhrnnou informaci o vlivu snížení/zvýšení hodnoty příslušného parametru regulátoru na celkovou odezvu regulačního pochodu, a jako taková, je vhodným výchozím prostředkem k formulaci klasických fuzzy pravidel, popisujících vliv změn hodnot příslušných parametrů.

Článek [6] představuje stručný souhrn použití fuzzy inferenčních technik v oblasti regulace procesů. Článek je rozdělen do tří částí. První se zabývá fuzzy PI a fuzzy PD regulátorem a jejich porovnání s konvenčním PID regulátorem, druhá fuzzy PID paralelními strukturami a třetí fuzzy supervisingem PID regulátoru. Při návrhu fuzzy supervisora autoři navazují na článek [5]. Vstupem fuzzy supervisora jsou tři kvalitativní indikátory: 1) *přeregulování*, 2) *doba náběhu* a 3) *faktor stability* definovaný jako poměr prvního podregulování k prvnímu přeregulování. Fuzzy inferenční báze jsou použity k nastavování hodnot K_p, T_i a T_d parametrů PID regulátoru.

V konferenčním příspěvku [7] je uvažován supervising diskrétního PI regulátoru. Nastavovanými parametry jsou proporcionální zesílení K_p a integrální zesílení K_i. Jsou konstruovány dvě samostatné FIB, každá pro nastavování jednoho parametru. Vstupními veličinami obou FIB jsou 1) *přeregulování* a 2) *doba náběhu*. V článku je krátce diskutována otázka volby intervalu vzorkování pro supervisora s tím, že je volena proměnná doba intervalu vzorkování.

5. Kategorie IC

Tato kategorie reprezentuje články s poněkud ortodoxnějším přístupem k použití fuzzy inferenčních technik než kategorie předešlé. Pokud je výhodnost použití fuzzy-PID hybridních struktur oproti použití samostatných fuzzy regulátorů opravňována poukazem na zachování některých výhodných vlastností PID regulace, naskýtá se otázka, proč tento princip nezachovat v ještě větší míře. Ziegler-Nicholsova formule je asi nejznámějším algoritmem pro konstantní nastavení parametrů PID regulátoru. Její použití je prakticky dobře prověřeno v široké třídě případů a je více či méně adekvátní. Přístup zvolený v článcích této kategorie vychází z myšlenky používat nastavení parametrů podle ZN-formule jako primární, ovšem toto nastavení na základě fuzzy inference vhodně dynamicky modifikovat, za účelem celkového zlepšení regulačního pochodu. Větší ortodoxnost přístupu tkví v tom, že pokud můžeme relativně jednoduše zjistit nějakou informaci o regulovaném procesu, v případě použití ZN-formule hodnoty kritického zesílení K_u a kritické periody T_u, je účelné nezbavovat se použití této informace ve vlastním algoritmu řídící struktury.

Ve článku [8] čínských autorů je navrhováno parametrizovat ZN-formuli jedním parametrem α, ve tvaru:

$$K_p = 1.2\alpha K_u,\ T_i = 0.75\frac{T_u}{1+\alpha},\ T_d = 0.25T_i.$$

Pro hodnotu $\alpha = 0.5$ přechází parametrická formule v původni ZN-formuli. Parametr α je předmětem on-line nastavování pomocí FŘS. FŘS se vnitřně skládá z jedné FIB a rekurzivní rovnice. Vstupem FIB jsou hodnoty regulační odchylky a jejího inkrementu. Výstupní hodnotou FIB je pomocná fuzzy veličina H. Po obdržení ostré hodnoty $h(t)$ pomocné veličiny H je použita adaptační rekurzivní rovnice pro výpočet hodnoty parametru α ve tvaru:

$$\alpha(t+1) = \begin{cases} \alpha(t) + \gamma h(t)(1-\alpha(t)) & \text{pro} \quad \alpha > 0.5 \\ \alpha(t) + \gamma h(t)\alpha(t), & \text{pro} \quad \alpha \le 0.5 \end{cases},$$

kde γ je kladná konstanta řídící rychlost konvergence. Hodnota $\alpha(0)$ není libovolná, ale je nastavena na hodnotu $\alpha(0) = 0.5$. To znamená, že na začátku regulace je použito nastavení parametrů podle klasické ZN-formule.

Pravidla FIB stejně jako rekurzivní adaptační formule jsou navržena tak, aby realizovali následující koncept. Pokud se hodnota regulované veličiny blíží k žádané hodnotě je hodnota parametru α zvýšena, tím je zvýšena hodnota K_p a sníženy hodnoty T_i a T_d, což má za následek rychlejší konvergenci k žádané hodnotě. Pokud hodnota regulované veličiny diverguje od žádané hodnoty je naopak hodnota parametru α snížena, tím je snížena hodnota K_p a zvýšeny hodnoty T_i a T_d, což následně zpomaluje divergenci.

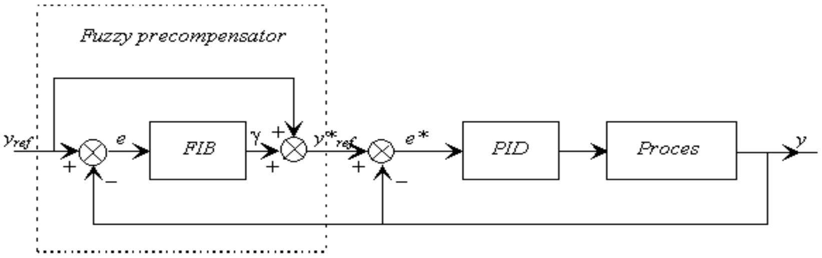
Článek [9] vychází z myšlenky korekce hodnot, jednotlivých, dle ZN-formule nastavených parametrů PID regulátoru, prostřednictvím přídavného aditivního členu ve tvaru:

$$\begin{aligned} K_p &= K_p + FIB(e(k), \Delta e(k)) \times k_1, \\ T_i &= T_i + FIB(e(k), \Delta e(k)) \times k_2, \\ T_d &= T_d + FIB(e(k), \Delta e(k)) \times k_3. \end{aligned}$$

Korekční aditivní člen $FIB(e(k), \Delta e(k))$ je výstupem fuzzy inferenční báze, jejímiž vstupy jsou hodnoty regulační odchylky a jejího inktrementu. k_1, k_2 a k_3 jsou konstanty určující váhu změn aplikovaných na jednotlivé parametry.

6. Fuzzy precompensator

Tuto kategorii reprezentuje článek [10]. Autoři tohoto článku, zvolily ke kombinaci FŘS a PID regulátoru jiný přístup než supervising. Při supervisingu jsou hodnoty parametrů PID regulátoru dynamicky měněny, a tím je ovlivňována jeho činnost. Druhou možnou cestou jak jeho činnost ovlivnit je dynamická úprava hodnot regulační odchylky a jejího inkrementu. Regulátor pak vychází z těchto upravených hodnot. Změna regulační odchylky se děje prostřednictvím změny žádané hodnoty. Parametry regulátoru jsou v tomto případě nastaveny pomocí nějaké konvenční metody a jsou v průběhu regulačního pochodu konstantní. Při tomto způsobu kombinace FŘS a PID regulátoru, fuzzy řídící strukturra vystupuje jako tzv. precompensator. Navrhovaný koncept blíže ozřejmí schéma regulačního obvodu na obrázku 1.



Obrázek 1: Schéma regulačního obvodu.

Vnitřně je fuzzy precompensator tvořen jednou FIB. Vstupy FIB tvoří skutečné hodnoty regulační odchylky $e(k) = y_{ref} - y(k)$ a jejího inkrementu $\Delta e(k) = e(k) - e(k-1)$. Výstupem FIB je fuzzy veličina Γ jejíž defuzzifikovaná hodnota $\gamma(k)$, představuje korekční aditivní člen použitý k úpravě žádané hodnoty regulované veličiny na tvar: $y^*_{ref}(k) = y_{ref} + \gamma(k)$. Jak je vidět na obrázku 1, je pak na základě této zdánlivé žádané hodnoty vypočítávána hodnota zdánlivé regulační odchylky $e^*(k) = y^*_{ref}(k) - y(k)$ a jejího inkrementu $\Delta e^*(k) = e^*(k) - e^*(k-1)$. Teprve hodnoty $e^*(k)$ a $\Delta e^*(k)$ tvoří vlastní vstupní veličiny PID regulátoru.

V souvislosti s článkem [10] je zajímavý článek [11], který se zabývá automatickým generováním fuzzy pravidel pomocí genetických algoritmů. Navrhovaný způsob generování je testován právě na generování znalostní báze pro fuzzy precompensated regulaci.

7. Závěr

Na základě rozboru získaných článků bylo zjištěno, že uvedené přístupy k návrhu fuzzy-PID hybridních struktur lze rozdělit do dvou hlavních kategorií. Jednu představuje fuzzy supervising parametrů PID regulátoru a druhou fuzzy precompensated regulace PID regulátorem. Většina autorů se v současné době kloní k fuzzy supervisingu. Navrhované FŘS se liší jak počtem tak i vnitřním uspořádáním FIB. U téměř všech článků je efektivita regulace pomocí navrhovaných FŘS porovnávána s efektivitou regulace pomocí konvenčně nastavených PID regulátorů, vždy je konstatováno, že efektivita regulace v prvním případě je lepší než v případě druhém. Tento fakt ukazuje, že koncept fuzzy-PID hybridních struktur má své opodstatnění. Za vhodný směr dalšího výzkumu považuji nalezení vhodné platformy na základě které, by bylo možné jednotlivé přístupy mezi sebou porovnat a tak zjistit, který z nich je vhodnější pro návrh reálných regulátorů.

References

[1] Z.Y.Zhao, M.Tomizuka, S.Isaka: "Fuzzy gain scheduling of PID Controllers", *IEEE Transactions on Systems, Man and Cybernetics, Vol. 23, No. 5, pp.1392-1398, 1993.*

[2] Ch.Ling, T.F.Edgar: "A new fuzzy gain scheduling algorithm for process control", *American Control Conference, Chicago, U.S.A., 1992.*

[3] G.Mc.Milan, W.K.Wojsznis, G.T.Borders: "Flexible gain scheduler", *ISA Transactions 33, pp. 35-41, 1994.*

[4] G.V.S.Raju, J.Zhou: "Fuzzy logic adaptive algorithm to improve robustness in a steam generator level controller", *Control Theory and Advanced Technology, Vol. 8, No. 3, pp. 479-493, 1992.*

[5] J.Litt: "An expert system to perform on-line controller tuning", *IEEE Control System, Vol. 11, No. 3, 1991.*

[6] R. Ketata, D.D. Geest, A. Titli: "Fuzzy controller: Design, evaluation, parallel and hierarchical combination with PID controller", *Fuzzy Sets and Systems, Vol. 71, pp. 113-129, 1995.*

[7] P. Oliveria, P. Lima, J. Sentierio: "Fuzzy supervision on intelligent control systems", *European Control Conference, Grenoble, France, 1991, pp. 1226-1231.*

[8] S.Z. He, S. Tan, F.L. Xu, P.Z. Wang: "Fuzzy sulf-tuning of PID controllers", *Fuzzy Sets and Systems, Vol. 56, pp. 37-46, 1993.*

[9] S. Tzafestas, N. Papanikopulos: "Incremental fuzzy expert PID control" *IEEE Transactions on Industrial Electronics, Vol. 37, No. 5, pp. 365-371, 1990.*

[10] J.-H. Kim, K.-C. Kim, E.K.P. Chong: "Fuzzy precompensated PID controllers", *IEEE Transactions on Control Systems Technology, Vol. 2, No. 4, pp. 406-410, 1994.*

[11] H.-J. Cho, K.-B. Cho, B.-H. Wang: "Fuzzy-PID hybrid control: Automatic rule generation using genetic algorithm", *Fuzzy Sets and Systems, Vol. 92, pp. 305-316, 1997.*

Numerické modelování některých případů kontaktního problému

doktorand:
Mgr. Zdeněk Kestřánek
ÚI AV ČR, Pod vodárenskou věží 2, Praha 8, 182 07
zdenda@uivt.cas.cz

školitel:
doc. Ing. Jiří Nedoma, CSc.
ÚI AV ČR, Pod vodárenskou věží 2, Praha 8, 182 07
nedoma@uivt.cas.cz

obor studia:
matematické modelování

Abstrakt

Problém, kdy alespoň jedno těleso zkoumané soustavy není úplně uchyceno, tj. příslušný funkcionál je semikoercitivní, se v kontaktní mechanice objevuje poměrně často. Příspěvek se zabývá numerickými metodami pro tyto případy. Klasická Uzawova metoda nekonverguje pro třídu problémů dostatečně rychle, a proto zde byla snaha využít jiné přístupy. Fyzikální povaha problému nabízí transformovat původní problém na posloupnost problémů regulárních (tj. všechna tělesa jsou pevně uchycena). Tvar těchto regulárních problémů nám potom dovoluje využít téměř standardní algoritmy k řešení. Některé úpravy, týkající se například takových otázek, jako je přechod mezi jednotlivými regulárními problémy a rotace kontaktních stupňů volnosti, však byly nezbytné.

1. Úvod

V kontaktní mechanice často přicházíme k modelům, kdy alespoň jedno těleso soustavy není pevně uchyceno, tj. jedná se o modely semikoercitivní. Soustřeďujeme se na takové semicoercitivní problémy, u kterých existuje jednoznačné řešení. Jako nejjednodušší možnost k řešení se nabízí Uzawův algoritmus sedlového bodu [3], ačkoliv v literatuře jsou uváděny konvergenční věty pouze pro případy, kdy energetický funkcionál je koercitivní na celém prostoru. K efektivnější implementaci této metody by bylo nutné provést "před-eliminační" krok. Toto ovšem vede k narušení šířky pásu matice tuhosti. Varianty založené na rozkladu Lagrangiánu jsou komplikovány dvěma skutečnostmi při vkládání a vyřazování omezujících podmínek do množiny aktivních omezení. Předně je nutné obnovovat faktor efektivně a za druhé, šířka pásu musí být zachována malá. Poznamenejme, že v semikoercitivních případech nemůžeme odděleně rozložit matici tuhosti.

Rozšířili jsme duální techniku [4], kdy třecí člen je již zahrnut do funkcionálu, na obecný případ kontaktu pružných těles. V příspěvku budou diskutovány též implementační aspekty týkající se této techniky.

2. Algoritmy

V diskrétní podobě vede kontaktní problém na problém [5] :

Nalézt $x \in K_d$ takové, že

$$\tilde{f}(x) \leq \tilde{f}(y) \quad \forall\, y \in K_d, \tag{1}$$

kde

$$\tilde{f}(y) = \tfrac{1}{2} y^T C y - y^T d + \sup_{|\mu_i| \leq 1} (y^T G^T \mu),$$

$$K_d = \{y \in R^N | Ay \leq 0\}\,.$$

Jelikož funkcionál $\tilde{f}(y)$ je nediferencovatelný, je výhodné převést problém (1) na problém sedlového bodu. Tím je problém v jistém smyslu "linearizován". Jsou možné dvě formulace sedlového bodu. V první je odstraněna pouze nediferencovatelnost, ve druhé jsou navíc do Lagrangiánu zahrnuty podmínky nepronikání. První formulace je řešena Uzawovým algoritmem [2, 3]. Odvoďme formulaci druhou. Definujme

$$L = \{\mu \in R^{M+P} \,|\, \mu_i \geq 0,\ i = 1, \ldots, M;\ |\mu_i| \leq 1,\ i = M+1, \ldots, M+P\}$$

a $B = \begin{pmatrix} A \\ G \end{pmatrix}$.

Potom hledáme sedlový bod

$$\bar{f}(y, \mu) = \tfrac{1}{2} y^T C y - y^T d + y^T B^T \mu$$

na množině $R^N \times L$. To lze přepsat jako :

Nalézt dvojici $(x, \lambda) \in R^N \times L$ takovou, že

$$Cx = d - B^T \lambda \tag{2}$$

$$x^T B^T (\mu - \lambda) \leq 0 \qquad \forall \mu \in L. \tag{3}$$

Lze dokázat [4] ekvivalenci mezi (1) a (2)-(3). V případech, kdy C je singulární, vznikají nesnáze. U kontaktních problémů se tyto případy vyskytují poměrně často. Práce [1] řeší takový případ užitím pseudoinverzní matice. Zde vycházíme z práce [4], která je založena vlastně na podobných principech. V [4] je řešen kontakt pružného tělesa s tuhou podložkou. Zde je tento případ rozšířen na obecnější kontakt pružných těles. Hlavní myšlenka tkví v oddělení některých neznámých v (2)-(3), které přísluší k určitým kontaktním podmínkám nepronikání.

3. Regularizace matice tuhosti

Předpokládáme, že pro každé těleso soustavy existuje kontaktní podmínka, která určuje indexy k, l takové, že 1) pro prvek a_{lk} matice A platí $a_{lk} \neq 0$ a 2) neznámá k přísluší uzlu vyšetřovaného tělesa. Navíc předpokládáme, že existuje taková lokální souřadná soustava, že dvojice (k, l) je určena jednoznačně.

Uvažujme problém (2)-(3). První a druhou složku řešení tohoto problému budeme po řadě nazývat primární a duální složkou. Zvolíme hodnoty pro primární složky a vypočítáváme složky duální ze vztahu, který bude důsledkem následujícího:

$$z^T d - z^T B^T \lambda = 0 \qquad \text{pro } z \in Ker\ C. \tag{4}$$

Proto pro všechna tělesa soustavy oddělujeme tolik kontaktních podmínek, kolik je dimenze příslušné množiny tuhých posunutí (obvykle 0 nebo 1). Symboly k a l budeme označovat primární a duální indexy, které přísluší odděleným kontaktním podmínkám. Předpokládáme, že upevnění primárních složek je popsáno substitucí

$$x_k = \alpha_k + \gamma x_p, \tag{5}$$

kde $\gamma = 1$ pro kontakt pružných těles a x_p odpovídá protějšímu uzlu v kontaktní dvojici, $\gamma = 0$ pro kontakt s tuhou podložkou.

Matice B má tvar:

$$B^T = (b_1, \ldots, b_l, \ldots, b_M, \ldots, b_{M+P}),$$

kde $b_l = (0, .., n_k, .., -\gamma n_k, .., 0)$, $n_k = a_{lk}$. Nechť navíc

$$B_0^T = (b_1, \ldots, 0, \ldots, b_M, \ldots, b_{M+P}), \qquad B_1^T = (b_l), \qquad \tilde{B}_1^T = (\tilde{b}_l),$$

kde $\tilde{b}_l = (0, .., n_k, .., 0)$. Potom (3) je ekvivalentní

$$x^T B_0^T(\mu - \lambda) \leq 0 \qquad \& \qquad x^T B_1^T(\mu - \lambda)_l \leq 0. \tag{6}$$

Pravá nerovnost je upravována dále:

$$x^T B_1^T \lambda_l = 0, \qquad x^T B_1^T \leq 0, \qquad \lambda_l \geq 0 \qquad \Leftrightarrow \tag{7}$$

$$-\lambda_l^T B_1(y - x) \geq 0 \qquad \forall y \quad B_1 y \leq 0. \tag{8}$$

Nechť $Tw \equiv w + \rho\lambda_l$, kde w je libovolný vektor stejné dimenze jako λ_l a $\rho > 0$. Potom (8) je ekvivalentní

$$\begin{aligned} (B_1 x - T(B_1 x))^T (B_1 y - B_1 x) \quad &\geq \quad 0 \quad \Leftrightarrow \\ B_1 x = -(T B_1 x)^- & \end{aligned} \tag{9}$$

V případě, kdy je oddělována pouze jedna kontaktní podmínka, (9) se redukuje na

$$x_k n_k = -(x_k n_k + \rho\lambda_l)^- \tag{10}$$

pro případ "tuhého" kontaktu.

Z předchozích úvah, společně s (5), vyplývá následující iterační algoritmus:

$$Cx^n = d - B^T \lambda^n \tag{11}$$

$$(x^n)^T B_0^T(\mu - \lambda^n) \leq 0 \qquad \forall \mu \in L \tag{12}$$

$$\tilde{B}_1 \alpha^{n+1} = -(\tilde{B}_1 \alpha^n + \rho\lambda_l^n)^- \tag{13}$$

Protože v každé iteraci předepisujeme hodnoty pro primární složky, definujme

$$C_0 = \begin{cases} c_{ij} & ((i \neq k) \wedge (j \neq k)) \vee (i = j = k) \\ 0 & ((i = k) \vee (j = k)) \wedge (i \neq j) \end{cases}$$

$$d_0 = \begin{cases} d_i & i \neq k \\ 0 & i = k \end{cases}$$

Nechť nyní pro jednoduchost η označuje matici jejíž sloupce tvoří bázi pro jádro C. Pak podobně jako u (4), plyne z (11) to, že můžeme určit hodnoty oddělených duálních složek:

$$(-\eta^T B_1^T)\lambda_l^n = \eta^T(d - B_0^T \lambda^n). \tag{14}$$

U jedné oddělené podmínky je člen $\eta^T B_1^T$ nenulové číslo. Pokud uvažujeme, že substituce (5) je přenesena do pravé strany vektorem d_α a $\tilde{B}_1 \alpha^0 \leq 0$, algoritmus (11)-(13) se mění na

$$C_0 x^n = d_0 + d_\alpha - B_0^T \lambda^n \tag{15}$$

$$(x^n)^T B_0^T(\mu - \lambda^n) \leq 0 \qquad \forall \mu \in L \tag{16}$$

$$\lambda_l^n = (-\eta^T B_1^T)^{-1} \eta^T (d - B_0^T \lambda^n) \tag{17}$$

$$\tilde{B}_1 \alpha^{n+1} = -(\tilde{B}_1 \alpha^n + \rho\lambda_l^n)^- \tag{18}$$

VĚTA. Pro dostatečně malé ρ algoritmus (15)-(18) konverguje k řešení (2)-(3).
Důkaz: Viz [5].

References

[1] Z. Dostál, "Duality based domain decomposition with proporting for the solution of free boundary problem", *J.Comp.Appl.Math.*, vol. 63, pp. 203–208, 1995.

[2] I. Hlaváček, J. Haslinger, J. Nečas, J. Lovíšek, *"Riešenie variačných nerovností v mechanike"*, Alfa, Bratislava, 1982.

[3] J. Horák, H. Netuka, "Numerical realization of contact problem with friction - semicoercive case", in: *Proc.Int.Conf. Mathematical Methods in Engineering*, Czech.Sci.Techn.Soc. ŠKODA Concern, Plzeň, pp. 147–152, 1991.

[4] D. Janovská, *"Numerical solution of contact problems"*, Dissertation, MFF UK, Praha, 1980.

[5] Z. Kestřánek, Numerical Modelling of Some Particular Cases in Contact Problem, in: M. Feistauer, R. Rannacher and K. Kozel, Eds., *Proc. 3rd Summ.Conf. NMICM'97*, Praha, pp. 321–328, 1997.

[6] J. Nedoma, "Finite element analysis of contact problems in thermoelasticity. The semi-coercive case", *J.Comp.Appl.Math.* vol 50, pp. 411–423, 1994.

Numerické řešení duální kontaktní úlohy v elasticitě

doktorand:
Mgr. Marek Novický

Muchova 13, 160 00, Praha 6

mano@uivt.cas.cz

školitel:
Doc.Ing. Jiří Nedoma, Csc.

Ústav informatiky AV ČR, Pod Vodárenskou věží 2,
Praha 8, 182 07

nedoma@uivt.cas.cz

obor studia:
M11 - Vědeckotechnické výpočty

Abstrakt

V první části článku je diskutována matematická analýza duální kontaktní úlohy. Variační formulace dulální úlohy je odvozena z primární úlohy použitím teorie sedlového bodu. Druhá část článku je zaměřena na numerické řešení úlohy metodou konečných prvků. Výsledky jsou prezentovány na jednoduchém modelovém příkladu.

1. Primární kontaktní úloha

Nechť dvě elastická tělesa zaujímají ohraničené oblasti $\Omega', \Omega'' \subset \mathrm{R}^2$ s lipšicovkými hranicemi a nechť jsou v kontaktu na části hranice Γ_{K}. Na části hranice Γ_τ je předepsáno zatížení. Těleso Ω' je upevněno na části hranice Γ_{u}. Předpokládáme homogenní isotropní materiály, konstantní teplotu, nulové objemové síly a nulové tření na kontaktní hranici.

2. Duální kontaktní úloha

Definujme prostor parametrů

$$S = \{\mathcal{N} = (\mathcal{N}_{\rangle|}),\ \rangle, | = \infty, \in,\ \mathcal{N}_{\rangle|} \in \mathcal{L}^{\in}(\otimes),\ \mathcal{N}_{\rangle|} = \mathcal{N}_{|\rangle}\},$$

kde $\Omega = \Omega' \cup \Omega''$ a položme

$$\mathcal{N}_{\rangle|} = \rceil_{\rangle|}(\mathbf{v}). \tag{19}$$

Potom platí

$$\mathcal{L}(\mathbf{v}) = \frac{1}{2}\int\limits_{\Omega} c_{ijkm}\mathcal{N}_{\rangle|}\mathcal{N}_{\|\Updownarrow}\lceil\mathbf{x} - \mathcal{L}(\mathbf{v}) = \mathcal{L}_\infty(\mathcal{N}, \mathbf{v}). \tag{20}$$

Minimalizace functionálu $\mathcal{L}_1(\mathcal{N}, \mathbf{v})$ definovaného (20) s omezeními (19) je ekvivalentní variační formulaci primární úlohy.

Definujme následující Lagrangián

$$\mathcal{H}([\mathcal{N},\mathbf{v}],\lambda) = \mathcal{L}_\infty(\mathcal{N},\mathbf{v}) + \int\limits_{\otimes} \lambda_{\rangle|}(\rceil_{\rangle|}(\mathbf{v}) - \mathcal{N}_{\rangle|})\lceil\mathbf{x}. \tag{21}$$

Problém nalezení

$$\sup_{\lambda\in S}\ \inf_{[\mathcal{N},\mathbf{v}]\in\mathcal{S}\times\mathcal{K}}\ \mathcal{H}([\mathcal{N},\mathbf{v}],\lambda) \tag{22}$$

nazýváme variační formulací duální úlohy.

Lze dokázat, že platí

$$\inf_{[\mathcal{N},\mathbf{v}]\in\mathcal{W}} \mathcal{H}([\mathcal{N},\mathbf{v}],\lambda) = \begin{cases} -\mathcal{J}(\lambda) & \text{for } \lambda\in K_L^+, \\ -\infty & \text{for } \lambda\notin K_L^+, \end{cases}$$

kde

$$\mathcal{J}(\lambda) = \frac{1}{2}\int\limits_{\Omega} a_{ijkm}\lambda_{ij}\lambda_{km}d\mathbf{x}. \tag{23}$$

$$K_L^+ = \{\lambda\in S|\ \mathcal{H}_2(\mathbf{v},\lambda)\geq 0 \quad \forall\mathbf{v}\in K\},$$

a kde a_{ijkm} jsou koeficienty inverzního Hookova zákona.

Tedy platí

$$\sup_{\lambda\in S}\inf_{[\mathcal{N},\mathbf{v}]\in\mathcal{W}} \mathcal{H}([\mathcal{N},\mathbf{v}],\lambda) = \sup_{\lambda\in\mathcal{K}_{\mathcal{L}}^+}[-\mathcal{J}(\lambda)] = -\inf_{\lambda\in\mathcal{K}_{\mathcal{L}}^+}\mathcal{J}(\lambda).$$

Z teorie sedlového bodu plyne, že pokud existuje sedlový bod funkcionálu $\mathcal{H}$, potom platí

$$-\inf_{\lambda\in K_L^+}\mathcal{J}(\lambda) = -\mathcal{J}(\tau(\mathbf{u})) = \mathcal{L}(\mathbf{u}), \tag{24}$$

kde $\mathbf{u}$ značí řešení primární úlohy.

Na druhé straně, existence řešení primární úlohy implikuje existenci sedlového bodu.

Nutné podmínky existence řešení $\mathbf{u}$ primární úlohy zaručují, že rozdíl dvou libovolných řešení patří do množiny posunutí tuhých těles, z čehož plyne, že tenzor deformace $e(\mathbf{u})$ a tenzor napětí $\tau(\mathbf{u})$ jsou určeny jednoznačně.

3. Metoda konečných prvků

Pro numerické řešení duální úlohy byl zvolen rovnovážný model metody konečných prvků podle Watwooda a Hertze.

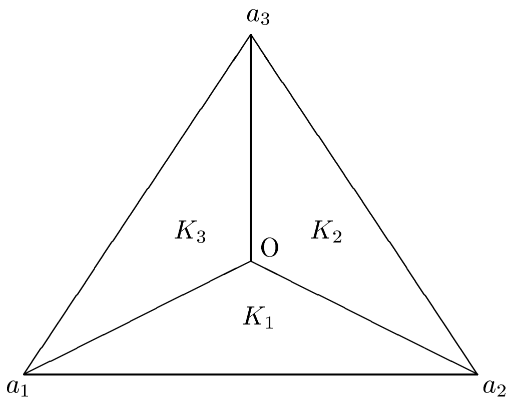
Blokový element K se sestaven ze tří podtrojúhelníků, které vzniknou z trojúhelníka spojením vrcholů s těžištěm (viz obr. 3).

Sestavením matice tuhosti z funkcionálu $\mathcal{J}$ vznikne matice $\mathbf{F}$ – symetrická matice $N\times N,\ N = 7 * 3 * N_t$,$N_t$ – počet trojúhelníků triangulace, která je pozitivně definitní. Pro funkcionál $\mathcal{J}$ dostaneme

$$\mathcal{J}(\mathbf{z}) = \frac{1}{2}\mathbf{z}^T\mathbf{F}\mathbf{z}.$$

Z okrajových podmínek sestavíme matici $\mathbf{D}$, typu $M_1\times N$, M_1 – počet podmínek. Dostaneme

$$\mathbf{Dz} = \mathbf{g},$$



Obrázek 3: Schéma blokového elementu

kde **g** obsahuje nuly a konstanty předepsaného zatížení.

Z kontaktních podmínek sestavíme matici **E**, typu $M_2 \times N$, M_2 – počet podmínek. Dostaneme

$$\mathbf{Ez} \leq 0.$$

References

[1] NEDOMA J., *Finite-element Analysis of Contact Problems in Thermoelasticity* , Journal of Computational and Applied Mathematics 50 (1994) 411-423

[2] NEDOMA J., *On the Signorini Problem With Friction in Linear Thermoelasticity. The Quasi-coupled 2D-case*, Apl. mat. 32 (1987) 186-199

[3] HLAVÁČEK I.-HASLINGER J.-NEČAS J.-LOVÍŠEK J., *Riešenie varičných nerovností v mechanike*, Alfa, Bratislava (1982)

[4] NEČAS J.-HLAVÁČEK I., *Úvod do matematické teorie pružných a pružně plastických těles*, SNTL, Praha (1983)

[5] WATWOOD V. B.-HARTZ B. J., *An Equilibrium Stress Field Model for Finite Element Solutions of Two-dimensional Elastostatic Problems*, Int. J. Solid Struct. 4 (1968) 857-873

[6] KESTŘÁNEK Z., *Comparison of Methods for Solving Contact Problem in Thermoelasticity*, Numerical Methods in Continuum Mechanics, Proc. of the Internat. Scient. Conf., Tiskove centrum VSDS Žilina, Stará Lesná, Slovensko (1994) 128-135

[7] KESTŘÁNEK Z., *Numerical Analysis of the Contact Problem - Comparison of Methods for Finding the Approximate Solution*, TR-ICS 648, Praha (1995)

Aditivní rozklad funkce ve dvojím čase

doktorand:
RNDr. Jan Klaschka
ÚI AV ČR, Pod vodárenskou věží 2, 182 07 Praha 8
klaschka@uivt.cas.cz

školitel:
Radim Jiroušek
VŠE, KIZI, nám. W. Churchilla 4, 130 67 Praha 4
radim@vse.cz

obor studia:
Operační výzkum

Abstrakt

Práce vypráví o autorově dobrodružné cestě za některými základními poznatky teorie užitku ze 60. a 70. let.

1. Aditivní rozklad a teorie užitku

V teorii užitku se (mimo jiné) studuje, zda preference mezi alternativami, z nichž si lze vybírat, je možné vyjádřit reálnou funkcí, a pokud ano, zda taková funkce může mít nějaký jednoduchý tvar. Ptáme se například, jestli míra užitku N-tice komodit je dána součtem měr užitku přiřazených jednotlivým komoditám. (Každý snadno vymyslí příklad týkající se skladby oběda či kombinace bot, kalhot a košile, kdy takový model zjevně "nesedí".) Odtud se dostáváme k problematice existence aditivního rozkladu funkce.

Mějme množinu A, která je podmnožinou kartézského součinu $X_1 \times \cdots \times X_N$ nějakých množin $X_1, \ldots, X_N$ a funkci $f : A \to \Re$ (kde $\Re$ je množina všech reálných čísel). Řekneme, že funkce f *má aditivní rozklad*, pokud existují takové funkce $f_i : X_i \to \Re$, $i = 1, \ldots, N$, že pro každé $(x_1, \ldots, x_N) \in A$ platí

$$f(x_1, \ldots, x_N) = \sum_{i=1}^{N} f_i(x_i). \tag{25}$$

Dále řekneme, že funkce f *má ordinálně aditivní rozklad*, pokud existuje taková rostoucí funkce $T : \Re \to \Re$, že složená funkce $g = Tf : A \to \Re$ má aditivní rozklad.

Z existence aditivního rozkladu funkce f zřejmě vyplývá platnost podmínky *kardinální nezávislosti*: Pokud $\mathbf{x}^1 = (x_1^1, \ldots, x_N^1), \ldots, \mathbf{x}^m = (x_1^m, \ldots, x_N^m)$ a $\mathbf{y}^1 = (y_1^1, \ldots, y_N^1), \ldots, \mathbf{y}^m = (y_1^m, \ldots, y_N^m)$ jsou takové prvky množiny A, že

$$\begin{array}{ccc} x_1^1, \ldots, x_1^m & \text{je permutací} & y_1^1, \ldots, y_1^m, \\ \vdots & \vdots & \vdots \\ x_N^1, \ldots, x_N^m & \text{je permutací} & y_N^1, \ldots, y_N^m, \end{array} \tag{26}$$

pak platí

$$\sum_{i=1}^{m} f(\mathbf{x}^i) = \sum_{i=1}^{m} f(\mathbf{y}^i).$$

Stejně triviálním důsledkem existence ordinálně aditivního rozkladu je podmínka *ordinální nezávislosti*: Pokud $\mathbf{x}^1 = (x_1^1, \ldots, x_N^1), \ldots, \mathbf{x}^m = (x_1^m, \ldots, x_N^m)$ a $\mathbf{y}^1 = (y_1^1, \ldots, y_N^1), \ldots, \mathbf{y}^m = (y_1^m, \ldots, y_N^m)$ jsou takové prvky množiny A, že platí (26), pak není možné, aby

$$f(\mathbf{x}^1) \geq f(\mathbf{y}^1), \ldots, f(\mathbf{x}^m) \geq f(\mathbf{y}^m),$$

kde aspoň jedna nerovnost je ostrá.

Obrácené implikace už tak zřejmé nejsou. Dá se s nimi dokonce zažít dobrodružství, jak se pokusím ukázat.

2. Od psychiatrie k teorii užitku

Koncem 80. let jsem, částečně ve spolupráci s profesorem Vopěnkou, studoval coby statistik pracující v psychiatrickém výzkumu nelogické vlastnosti jisté metody hodnocení zlepšení stavu pacienta. Navrhl jsem dvě (vlastně více, ale na tom teď nezáleží) varianty kritéria "rozumnosti" škál pro měření změny, tzv. regularity indexu změny a snažil jsem se způsoby hodnocení změny, které z hlediska kritérií regularity obstojí, charakterizovat. (Mluvil jsem o tom na minulém doktorandském dni.)

Kritérium tzv. *kardinální regularity* lze formulovat (i když "kanonická" formulace je trochu jiná – obsahuje pravděpodobnosti) jako podmínku kardinální nezávislosti pro jistou reálnou funkci f definovanou na podmnožině A kartézského čtverce $X \times X$ nějaké množiny. Analogický vztah je také mezi tzv. *ordinální regularitou* a podmínkou ordinální nezávislosti.

Pro kardinální regularitu jsme s profesorem Vopěnkou měli určité výsledky – uměli jsme dokázat, že za jistých dodatečných předpokladů splňuje funkce f definovaná na konečné množině $A \subseteq X \times X$ podmínku kardinální regularity, právě když existuje funkce $g : X \to \Re$ taková, že pro $(x_1, x_2) \in A$ platí $f(x_1, x_2) = g(x_1) - g(x_2)$.

Snažil jsem se dokázat, že ordinální regularitu lze charakterizovat obdobně, totiž že (opět za dodatečných předpokladů) funkce f splňuje podmínku ordinální regularity, právě když pro $(x_1, x_2) \in A$ platí $f(x_1, x_2) = T\big(g(x_1) - g(x_2)\big)$, kde g je reálná funkce definovaná na X a $T : \Re \to \Re$ je rostoucí.

"Masíroval" jsem problém zleva zprava – odolával. Věděl jsem, že už mohl být dříve vyřešen, poučení z literatury jsem se nijak programově nevyhýbal, ale nebylo jasné, kde (ve které disciplíně) a jak (pod jakými hesly) hledat.

Pomohla šťastná náhoda. Ukázal jsem svou hypotézu v jakési algebraické podobě dr. Karenovi, o němž se ne náhodou mluví jako o renesanční osobnosti. Něco mu to připomínalo, konkrétně tvrzení, že ke každé tzv. komparativní pravděpodobnosti existuje kvantitativní pravděpodobnost, která ji reprezentuje. (Není teď důležité, co to znamená.) To ovšem neplatí, protipříklad je v "modré knize" [4]. Protože se mi nepodařilo přesně převést svůj problém na problém uvedený v knize, resp. z protipříkladu v knize udělat protipříklad ke své hypotéze, vyhledal jsem dále článek [5], v knize citovaný, kde je charakterizace komparativních pravděpodobností reprezentovaných pravděpodobnostními mírami. Ukázalo se, že v článku, tehdy cca 25 let starém, jsou vyřešeny tři různé problémy, a že řešení jednoho z nich (nikoli ovšem toho, který se týká komparativní pravděpodobnosti, nýbrž problému tzv. uspořádaných diferencí) potvrzuje mou hypotézu.

V článku [5] jsem tedy místo protipříkladu nalezl důkaz. Co ale bylo neméně důležité, zjistil jsem konečně, kde se studují problémy, jaké mě zajímaly. Vlastně, kde se dříve studovaly…

3. Malé letní dobrodružství, aneb od teorie užitku málem na psychiatrii

Scott v [5] dokázal, že pro $A \subseteq X_1 \times X_2$ konečnou vyplývá z ordinální nezávislosti existence ordinálně aditivního rozkladu pro funkci $f : A \to \Re$ – neformuluje to sice jako větu, ale je to "mezivýsledek" obsažený v důkazu věty.

Přes Scottův článek jsem se dostal k množství další literatury o teorii užitku. Dříve než jsem si stačil půjčit první knihy a časopisy, jsem však použil techniku odpozorovanou u Scotta k důkazu tvrzení, že pro funkci f definovanou na konečné množině $A \subseteq X_1\ X_2$ vyplývá z podmínky kardinální nezávislosti existence aditivního rozkladu.

Zkráceně lze důkaz popsat takto: Bez újmy na obecnosti můžeme předpokládat, že množiny X_1 a X_2 jsou disjunktní. (Pokud nejsou, můžeme za cenu složitějšího značení pracovat s jejich "disjunktními kopiemi".) Označme X^* množinu $X_1 \cup X_2$ a $\mathcal{X}^*$ vektorový prostor všech reálných funkcí definovaných na X^*. Prvek $x \in X^*$ lze přirozeně ztotožnit s jeho charakteristickou funkcí $x^* \in \mathcal{X}^*$ a prvek $\mathbf{x} = (x_1, x_2) \in X_1 \times X_2$ s charakteristickou funkcí $\mathbf{x}^*$ množiny $\{x_1, x_2\}$ (takže $\mathbf{x}^* = x_1^* + x_2^*$). Nechť $A^* \subseteq \mathcal{X}^*$ je množina všech "protějšků" prvků množiny A a B^* je maximální lineárně nezávislá podmnožina A^*. Na B^* definujeme $f^*(\mathbf{x}^*) = f(\mathbf{x})$ a funkci f^* rozšíříme lineárně na $\mathcal{X}^*$. Dá se pak relativně snadno dokázat, že $f^*(\mathbf{x}^*) = f(\mathbf{x})$ platí pro všechna $\mathbf{x} \in A$. Pomocí funkce f^* definujeme funkce $f_i : X_i \to \Re$ $(i = 1, 2)$ předpisem $f_i(x) = f^*(x^*)$. Funkce f, f_1 a f_2 pak zřejmě (pro $N = 2$) na A splňují (25).

Potom jsem se seznámil s Fishburnovu monografií [2] z roku 1970, kde je problém existence aditivního rozkladu funkce (vlastně problém charakterizace tzv. aditivního středního užitku – ale to těsně souvisí) probrán velmi důkladně. Jedná se převážně o Fishburnovy vlastní výsledky. Pro konečnou množinu A, ať už v případě $N = 2$, nebo pro N obecné, lze větu o ekvivalenci existence aditivního rozkladu a kardinální nezávislosti dokázat jako malé cvičení z lineární algebry (a mezi cvičení, nikoli do hlavního textu, je také tato látka pro A konečnou v knize zařazena). Zato pro A nekonečnou (pokud $A \neq X_1 \times \cdots \times X_N$) je problém složitější. Fishburn v knize dokazuje, že existence aditivního rozkladu vyplývá z kardinální nezávislosti, jen pro $N = 2$ a tvrzení dokazuje pomocí grafového aparátu, který se nedá bezprostředně použít pro $N > 2$. O případu $N > 2$ říká, že není dosud uspokojivě vyřešen. Nepátral jsem nijak usilovně, zda byl vyřešen později (tj. po vydání knihy), protože pro záležitosti kolem psychiatrických škál byly relevantní jen výsledky týkající se případu $N = 2$.

Řadu let se pak nestalo nic, co by zde stálo za zmínku. Až letos v létě.

Prohlížel jsem si své staré poznámky včetně důkazu, že pro $N = 2$ a A konečnou je existence aditivního rozkladu ekvivalentní kardinální nezávislosti.

A najednou jsem to uviděl: Důkaz platí prakticky beze změny pro obecné N a A nekonečnou! Řeší tedy problém, který byl v knize [2] uveden jako otevřený. Přiznávám se, že mě to uvedlo do varu. Mohu to vůbec mít dobře, a může to jít tak jednoduše? Profesor Fishburn je přeci jen Někdo (kromě záplavy publikací, několika monografií a různých prestižních cen má také Erdösovo číslo – viz [1] – rovné jedné) a před vydáním knihy se problémem existence aditivního rozkladu funkce minimálně několik let zabýval.

Alarmoval jsem hned kamaráda z ÚTIA, kterému z dobrých důvodů v matematice věřím víc než sobě, abych s ním důkaz prošel a ověřil, že nedělám nějakou triviální chybu. Důkaz "držel". Tím jsem se pro změnu dostal do stavu trapné nejistoty: Nemám nakonec nový výsledek? Bylo to krajně nepravděpodobné – všichni měli na zdolání problému, dle knihy [2] otevřeného, skoro 30 let.

Po několika neúspěšných pokusech dozvědět se "kdo a kdy" z bibliografických databází jsem, nevida efektivnější způsob, třesoucí se rukou napsal profesoru Fishburnovi. Odpověděl brzy, mile a k věci. Problém vyřešil on a řešení publikoval v r. 1971 [3].

4. Dvojí čas, aneb zpět k realitě

Vylíčil jsem, jak se odkrývaly taje aditivních a ordinálně aditivních rozkladů funkcí v mém subjektivním čase.

Přejdeme-li k času objektivnímu, událo se toto: V 60. a 70. letech byly publikovány důležité výsledky teorie užitku. V 80. a 90. letech se s nimi postupně seznamoval jakýsi člověk z Prahy, který je potřeboval.

References

[1] R. de Castro, "O Erdösově čísle", *Pokroky matematiky, fyziky a astonomie*, vol. 43, pp. 130-135, 1998.

[2] P. C. Fishburn, *Utility Theory for Decision Making*, John Willey & Sons, New York, 1970.

[3] P. C. Fishburn, "Additive Representations of Real-Valued Functions on Subsets of Product Sets", *J. of Math. Psychology*, vol. 8, pp. 382-388, 1971.

[4] T. Havránek *et al.*, *Matematika pro biologické a lékařské vědy*, Academia, Praha, 1981.

[5] D. Scott, "Measurement structures and linear inequalities", *J. of Math. Psychology*, vol. 1, pp. 233-247, 1964.

Buněčná síť - model neuronové sítě s proměnlivou topologií

doktorand:
MGR. PŘEMYSL ŽÁK
ÚI AV ČR, Pod vodárenskou věží 2, 182 07 Praha 8
zak@uivt.cas.cz

školitel:
ING. MARCEL JIŘINA, DRSC.
ÚI AV ČR, Pod vodárenskou věží 2, 182 07 Praha 8
marcel@uivt.cas.cz

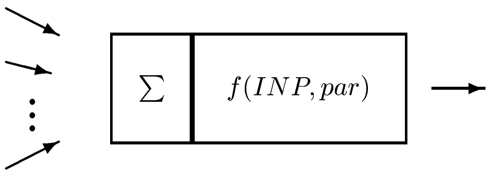
obor studia:
Výpočetní technika - Neuronové sítě

Abstrakt

Předmětem tohoto příspěvku je nový typ neuronové sítě inspirovaný poměry v imunitním systému. Základní výkonný prvek systému - buňka vyměňuje s ostatními signál stejně jako neuron. Buňka má však navíc schopnost pohybu, takže se její partneři v komunikaci mění. Navíc se buňka může i rozdělit, nebo zaniknout.

1. Základní vymezení

Buňka - Objekt se svými parametry a funkcemi. Jedním z mnoha parametrů buňky je její poloha. přijímá signál ze svého blízkého okolí, zpracuje ho a vyhodnotí jeho velikost.



Obrázek 4: Schéma buňky

V síti se pohybuje několik typů buněk. Každý typ má své specifické hodnoty některých parametrů.

Síť - obdélník $m\ x\ n$ možných pozic buněk. Buňky v každém kroku skončí na určitém místě. Na jednom místě však může být najednou i více buněk.

Signál - Každá buňka v každém kroku vyšle signál na základě vstupního signálu z minulého kroku a zároveň přijme vstupní signál nový. Vstupní signál je poscítán ze signálů buněk z nejbližšího okolí za použití vzdálenosti, vah a prahu. Na tento signál se pak aplikuje aktivační funkce buňky, kterou je jistá zvonovitá funkce. Na základě spočítané aktivity a se buňka dělí nebo zaniká. Aktivita buňky je pak vyjádřena v dalším kroku jako její výstupní signál.

Motivace - využití vlastností živých systémů k obohacení repertoáru neuronových sítí.

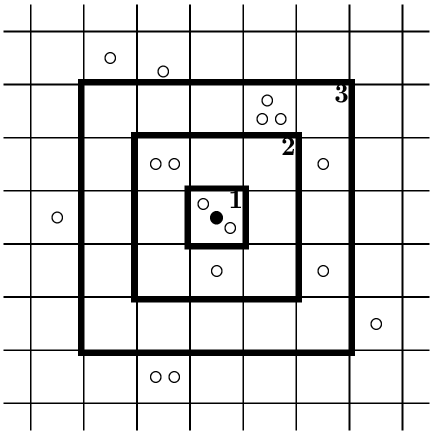
Cíl - Vytvořit stabilní systém se schopností učení.

2. Popis Buněčné sítě

Stavba sítě:

Celá síť je tvořena plochou (třeba čtvercem) skládající se z malých plošek (čtverečků), které představují místa výskytu buněk. Počet buněk přítomných najednou každém takovém místě je rovněž omezen.

Buňky se pohybují v jednotlivých krocích po síti. K každém kroku je prozkoumáno okolí buňky, z kterého může buňka přijmout signál.



Obrázek 5: Zóny působení buňky

V daném kroku buňka nejprve zpracuje signál pomocí vztahu

$$INP_i = \sum_{j=1}^{S} \frac{w_{i,j}}{l_{i,j}} X_{i,j} - THR_i \tag{27}$$

INP_i	*vstup i-té buňky*
S	*počet buněk v okolí*
$w_{i,j}$	*váha vztahu mezi i-tou a j-tou buňkou*
$l_{i,j}$	*vzdálenost mezi i-tou a j-tou buňkou*
$X_{i,j}$	*signál přicházející od j-té do i-té buňky*
THR_i	*práh i-té buňky*

Dále pak použije svou aktivační funkci. V praxi se nejlépe osvědčila zvonovitá funkce vzniklá odečtením dvou sigmoid, která svým tvarem nejlépe vystihuje biologickou skutečnost.

$$DSigm(INP, posunuti) = Sigm(INP) - Sigm(INP - posunuti) \tag{28}$$

Parametry sítě:
SÍŤ
I x J - rozměry sítě

K - maximální počet buněk na jednom místě
N - počet typů buněk
seed number - pro generátor pseudonáhodných čísel
temperature - maximální vzdálenost přesunu buňky
ampl - zesilovací konstanta signálu v síti

TYPY
name - název typu
max, steep - parametry aktivační funkce
alpha - koeficient rychlosti změn při učení
per - perioda působení učícího signálu
A - maximální věk buněk
supp - frekvence přísunu nových buněk
ND - perioda přirozeného dělení buněk
[0] - inicializační intervaly hodnot parametrů
W_i - váhový vektor

BUŇKA
ID - identifikační číslo buňky
age - věk
i, j - pozice buňky

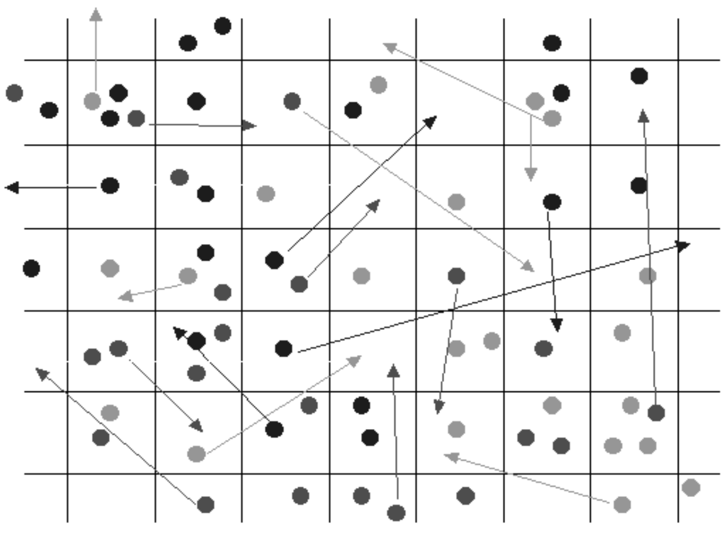
PARAMETRY UČENÍ (možné)
rec - práh buňky
mort, div - mez zániku a rozdělení buňky
lrn_i - individuální váhový vektor

Činnost sítě:
Nejprve se síť inicializuje. Nastavují se počáteční hodnoty parametrů platných pro celý model (seed, rychlost pohybu, parametry učení ...), platných pro jeden typ buněk (počet buněk v typu, limit pro dělení a zánik buněk, váhy ...) i pro jednotlivé buňky (identifikační číslo, poloha, věk ...).

Dalším krokem je příjem signálu z okolí buňky a jeho zpracování popsané výše.

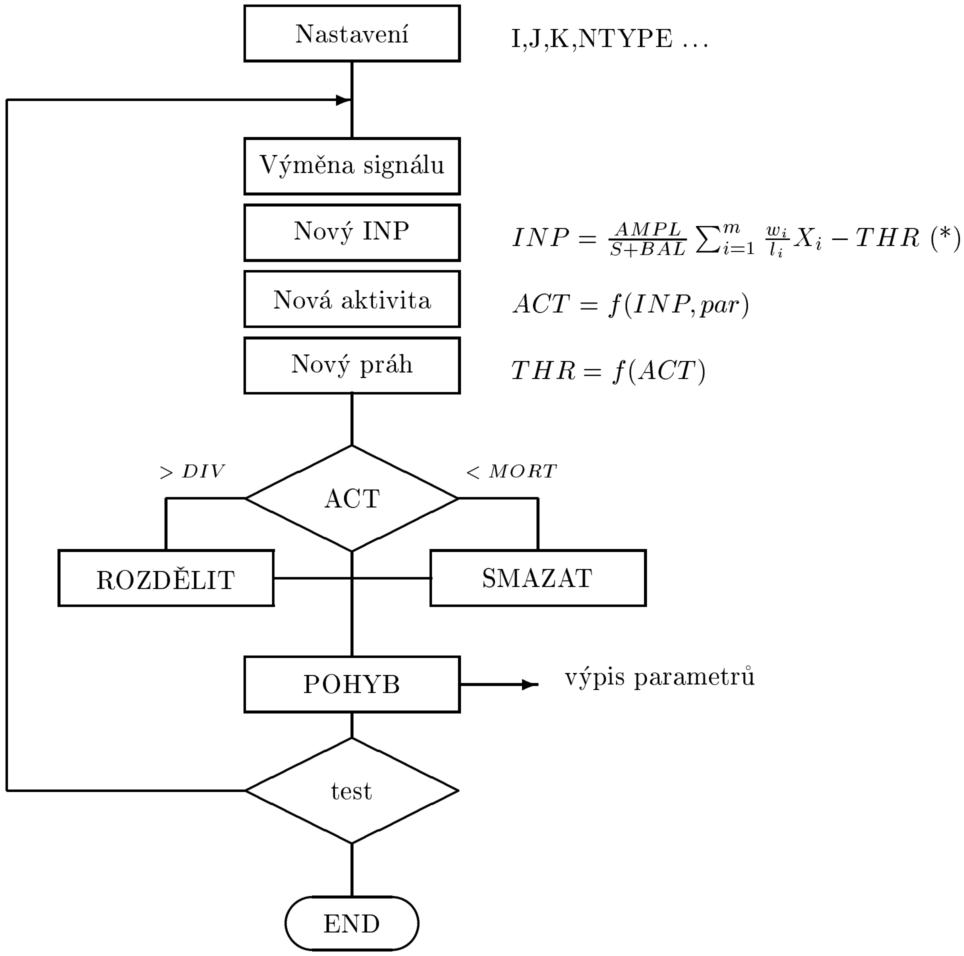
Výsledek tohoto zpracování se pak porovnává s mezemi pro dělení a zánik. Je li výsledná aktivita příliš malá, buňka zaniká, je-li naopak dostatečně vysoká, buňka se rozdělí. Překročí-li buňka svůj maximální věk, rovněž zaniká. Buňka se zde může též rozdělit přirozeným dělením, které probíhá náhodně s předem určenou četností.

Nakonec se buňky přemístí na své nové místo. Přesun je náhodný maximální vzdálenost přesunu je předem určena.



Obrázek 6: Schéma pohybu buněk po síti

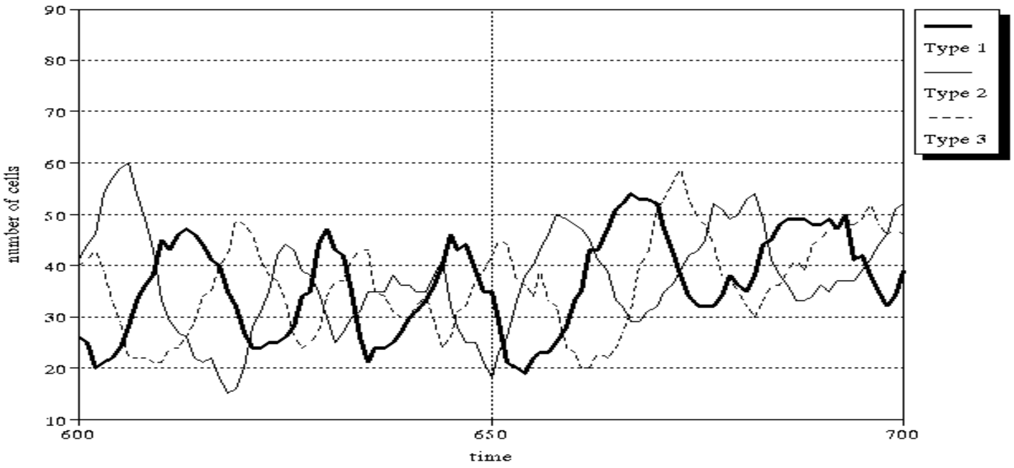
Celý proces lze stručně popsat tímto diagramem.



Obrázek 7: Schéma buněčné sítě

(*) S *je počet příjemců signálu v okolí buňky*
m *je počet aktivních buněk v síti*

V první fázi se podařilo nastavit model tak, aby byla stabilní. Počty buněk v jednotlivých typech se mění periodicky kolem nějaké střední hodnoty. V síti se pohybují buňky tří typů a váhy mezi nimi jsou nastaveny symetricky a činnost modelu připomíná systém lišky-zajíci.

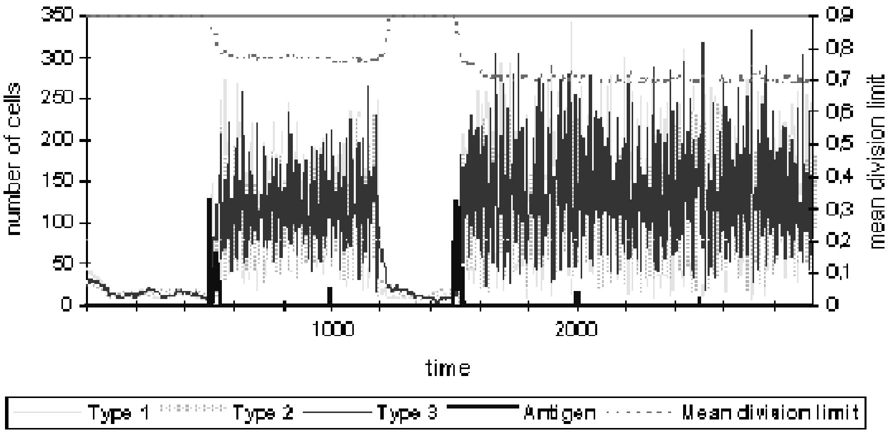


Obrázek 8: Detail činnosti modelu v klidu

Dále jsme se snažili vyvinout učící algoritmus sítě. Základem toho byla tato úvaha podpořená biologickou skutečností.

1. síť je v rovnováze
2. na buňky sítě působí jistý signál
3. síť reaguje a za čas t1 se ustaví nová rovnováha. Během této fáze probíhá učení.
4. nové působení téhož signálu. 5. ustavení rovnováhy v čase t2 « t1

Které parametry však měnit? Nejdříve jsme zkusili měnit váhový vektor buněk atakovaných signálem. Negativním efektem však byla poměrně rychlá ztráta stability sítě. Dále jsme měnili mez dělení. Po působení učícího signálu rychle početně převládly buňky se sníženým prahem dělení a po uplynutí jejich věku rázem zanikly. Jestliže jsme pak měnili mez zániku, naučení bylo dosti krátkodobé.



Obrázek 9: Ukázka chování sítě při zásazích učícího signálu

4. Závěr

Naše cíle byly splněny zatím jen částečně. Síť se chová jako stabilní systém a je schopná jakéhosi učení, které je zatím dosti nespecifické. Vyvstal však zároveň problém, jak tento model dále využít.

References

[1] Perelson, A.S.: Immune network theory. *Immunol.Rev.* **110**, 5-36 (1989)

[2] Stites, D.P., Terr, A.I.: *Basic and Clinical Immunology.* Applenton & Lange, A Publishing Division of Prentice Hall, (1991)

[3] DeBoer, R.J., Perelson, A.S., Kevrekidis, I.G.: Immune network behavior I. and II. *Bull. math. Biol.* **54** 745-816 (1993)

[4] *Pokroky v imunologii.* Univerzita Karlova, Sekce postgraduálního studia v biomedicíně (1994)

[5] Žák, P.: Immune Network - Model Inspired by Immune System. *Neural Network World*, Vol.7, **6**, 739-756 (1997)

[6] Žák, P.: Cell Network - Dynamic Neural Network. Learning Problems. *In: EUFIT'98. 6th Europien Congress on Intelligent Techniques and Softcomputing*, Vol.: 1. - Aachen, Mainz 1998, 323-325 (Held: EUFIT'98, Aachen, DE, 98.09.07-98.09.10)

Cascade Networks: Another Approach to Function Approximation

doktorand:
MGR. ARNOŠT ŠTĚDRÝ
Institute of Computer Science, Czech Academy of Sciences
Pod vodárenskou věží 2, 182 07, Prague, Czech Republic
arnost@uivt.cas.cz

školitel:
RNDR. JIŘÍ WIEDERMAN, DRSC.
Institute of Computer Science, Czech Academy of Sciences
Pod vodárenskou věží 2, 182 07, Prague, Czech Republic
wieder@uivt.cas.cz

obor studia:
Informatika

Abstrakt

A subclass of networks with cascade architecture is presented. We investigate its approximation capabilities by means of a continued fraction framework. It is shown that such network can approximate any meromorphic function arbitrarily well, which in the real domain covers even functions with second order discontinuity. Used techniques imply various alternative learning algorithms that are discussed.

1. Introduction

The most widely used and studied types of artificial neural networks have been multilayer perceptrons trained by various modifications of back–propagation algorithm. The learning process of such networks based on gradient descent in high-dimensional spaces is usually very time consuming. That is one of the reasons why cascade architectures of neural networks have appeared recently, together with incremental learning algorithms that gradually add new units to the network in order to achieve a better precision. In each step of these algorithms only the weights corresponding to added unit are changed while the rest of the weights is preserved. A typical example of such network architecture and learning algorithm is Fahlman's Cascor network (Fahlman, 1991). In many applications this network can learn reasonably faster in comparison with multilayer perceptrons. A theoretical background for incremental learning algorithms in multilayer perceptron networks which is based on Jones theorem concerning convergence of iterative approximation in a Hilbert space was laid by Barron, 1993.

It has been shown that multilayer perceptron networks with at least three layers are capable to represent any reasonable input/output function with arbitrary precision. This property, called *universal approximation*, is also true for various other architectures. This is why other criteria for judging the quality of the networks are being investigated. One of them might be the learning time, which seems to be a crucial problem especially in practical applications of neural networks.

In the following we consider architectures with richer topologies than classical multilayer perceptrons, particularly the ones in which connections are not limited only to units in neighboring layers. We call the architectures in which there are lateral connections between the neurons in one layer going from left to right *cascade architectures*.

In this article we introduce a subclass of cascade architectures and describe its approximation capabilities. It is shown that in complex domain any meromorphic function can be approximated arbitrarily well, which in real case includes even functions with the second order discontinuity. Moreover, rational functions are exactly representable by this architecture. The continued fraction calculus used in proofs provides new approaches to learning. In the following we briefly introduce two alternative learning algorithms.

2. Feedforward and Chain Architectures

By *multilayer perceptron* network we mean a network consisting of several layers of units (neurons) connected in a feedforward manner such that each unit in one layer has connections (synapses) with all units in the neighboring layers. It means that synapses between the preceding and the following layer forms a complete bipartite graph. Each connection has a real parameter, called weight, which is a subject of learning. The output y of one unit having inputs x_i with assigned weight values w_i and a threshold b is computed as

$$y = \sigma(\sum_{i=1}^{n} w_i x_i - b),$$

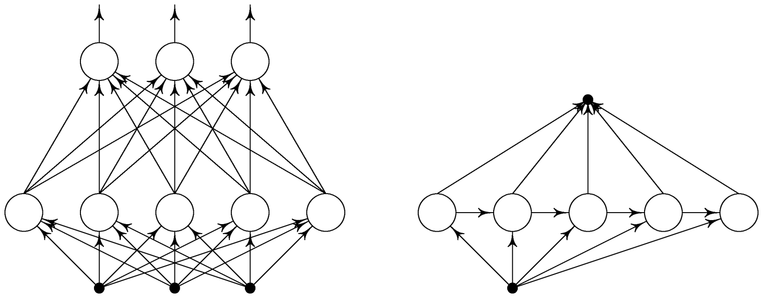
where $\sigma(t)$ is a sigmoidal function, i.e. non-decreasing function with the following limits: $\lim_{t\to\infty} \sigma(t) = 1$ and $\lim_{t\to-\infty} \sigma(t) = 0$. The most popular function—logistic sigmoid—writes as:

$$\sigma(t) = \frac{1}{1+\exp(-t)}$$

In the following, we consider a class of one-hidden-layer networks with a single linear output. The horizontal connections in the hidden layer connect the i-th unit with the $(i+1)$-th one (see Figure 10). The hidden units compute functions of the form:

$$o = \delta(z, y) = \frac{az+b}{cz+d+y}, \tag{29}$$

where y corresponds to the output of the preceding unit, z is the actual network input and a, b, c, d are complex parameters. We call a member of this subclass of cascade networks with the activation function of the form (29) a *chain network*. On the contrary to real numbers it is not clear which function should be chosen as an activation function of one unit. For example, there is no straightforward complex counterpart of the logistic sigmoid. That is the reason why various activation functions are used (cf. Hirose, 1992a, 1992b)). In our considerations we use the function of the form 29 which is widely used in the complex number theory and fits into the framework of continued fractions that will be used in the following. Furthermore, this function—as the simplest possible rational function—is very easy computable but it is still able to approximate discontinuities.



Obrázek 10: Multilayer perceptron and chain network

In order to formulate our results we need to define two classes of functions.

Definition 1 *By $\mathcal{F}$ we denote the set of all functions $f : \mathcal{C} \to \mathcal{C}$ representable by a chain network with any finite number of hidden units.*

Definition 2 *Denote $\mathcal{E}$ the set of functions $f(z) : \mathcal{C} \to \mathcal{C}$ of the form $\frac{p(z)}{q(z)}$, where p and q are polynomials.*

Finally, remind that a *homogeneous polynomial* of n variables $H_n(x_1, \ldots, x_n)$ has a form:

$$H_n(x_1, \ldots, x_n) = \sum_{j=1}^{m} a_j \prod_{i=1}^{n} x^{q_{ij}},$$

such that the sums $\sum_{i=1}^{n} q_{ij}$ are equal for every $j = 1 \ldots m$.

3. Continued Fractions

A function realized by chain network can be straightforwardly represented in the form of a continued fraction which formal definition immediately follows.

Definition 3 *By a* continued fraction *we mean the following formula:*

$$b_0 + \cfrac{a_1(z)}{b_1(z) + \cfrac{a_2(z)}{b_2(z) + \cfrac{a_3(z)}{b_3(z) + \ddots}}} \tag{30}$$

where a_i and b_i are complex domain functions of the variable z.

A partial continued fraction *$P_n(z)$ is an expression having only a finite number of coefficients a_i, b_i:*

$$P_n(z) = b_0 + \cfrac{a_1(z)}{b_1(z) + \cfrac{a_2(z)}{b_2(z) + \cfrac{a_3(z)}{\begin{array}{c}\vdots\\ b_{n-2}(z) + \cfrac{\vdots}{b_{n-1} + \cfrac{a_n(z)}{b_n(z)}}\end{array}}}}$$

If $\lim_{n\to\infty} P_n(z)$ exists and equals v, we say that continued fraction (30) has value *v.*

The framework of continued fractions represents a classical part of analytical calculus with basic results dating back to Euler, 1739. We introduce additional definitions and theorems here that are needed in the proof of the main theorem in the following section.

The main relation between a power series and a continued fraction describes the Euler identity (Danilov et al, 1961):

Theorem 4 (Euler) *If expressions on both sides exist then the following equation holds:*

$$\sum_{i=0}^{\infty} c_i z^i = \cfrac{c_0 + c_1 z}{1 - \cfrac{c_2 z}{c_1 + c_2 z - \cfrac{c_1 c_3 z}{c_2 + c_3 z - \ldots \ddots \vdots - \cfrac{c_{i-2} c_i z}{c_{i-1} c_i z} - \ddots}}}$$

The following definition and two theorems from Wall, 1948 show that a power series with negative exponents can be expanded as a continued fraction, as well.

Definition 5 *A sequence of polynomials $B_p(u) (p \leq m)$ is called* orthogonal relative to the sequence *c if*

$$\int B_p(u) B_q(u) d\phi_c(u) \begin{cases} = 0 \,;\, p \neq q, p, q \leq m \\ \neq 0 \,;\, p = q < m \end{cases}$$

Theorem 6 *A power series $\sum_{i=0}^{\infty}(c_i/z^{i+1})$ can be expanded into a continued fraction if and only if a sequence of polynomials can be constructed which are orthogonal relative to the sequence of coefficients $c_0, c_1, \ldots, c_n, \ldots$.*

Theorem 7 *Let $\sum_{i=0}^{\infty}(c_i/z^{i+1})$ be a power series and Δ_p determinants:*

$$\Delta_p = \begin{vmatrix} c_0 & c_1 & \ldots & c_p \\ c_1 & c_2 & \ldots & c_{p+1} \\ & \vdots & & \\ c_p & c_{p+1} & \ldots & c_{2p} \end{vmatrix},$$

such that:

1. *either $\Delta_p \neq 0$ for every p*
2. *or there is a number m: $\Delta_p \neq 0$ for $p \leq m$ and $\Delta_p = 0$ for $p > m$.*

Then a sequence of polynomials orthogonal relative to the sequence $\{c_0, c_1, \ldots\}$ can be constructed.

4. Main results

It is clear from the previous sections that there is a obvious correspondence between functions that are realized by a chain network and continued fractions. Exactly:

Fact 8 *Any function $f \in \mathcal{F}$ has the form of a partial continued fraction P_n, where n is the number of hidden units in the network.*

The following theorem says that functions realizable by chain networks can approximate any meromorphic function. It means that for any function and any precision we can find its approximator $f \in \mathcal{F}$, in another words—there is a network that, in principle, can learn this function arbitrarily well. Due to a limited space we only sketch the main ideas of the proof.

Theorem 9 *A chain network can approximate any meromorphic function with arbitrary precision. Moreover, function of class $\mathcal{E}$ can be exactly represented by a chain network.*

Sketch of proof. The typical way of proving universal approximation property is via the notion of density. In our case we prove that the set of functions $\mathcal{F}$ is dense in the space of meromorphic functions with the uniform metrics.

Any meromorphic function f can be expressed by means of a Laurent series, which is an infinite sum of the form

$$f(z) = \sum_{i=-\infty}^{\infty} a_i z^i = \sum_{i=0}^{\infty}(c_i/z^{i+1}) + \sum_{i=0}^{\infty} c_i z^i.$$

We can divide the sum into two parts according to the sign of exponent i. Now the problem is to realize both sums by continued fractions. Here we use two fundamental theorems that show how to realize the above two power series by continued fractions. The former can be expressed according to the Theorem 7, the latter is rewritten according to the Euler identity (4).

The last technical thing deals with obeying slight conditions of the Theorem 7 and Euler identity. As our series are of a general kind, we have to rewrite them as a sum of three series satisfying necessary presumptions, which finishes the proof.

So far, we have considered only chain networks with one-dimensional inputs and outputs, but the previous result can be extended for multiple-input networks as well. This is done by means of the dimension reduction, which is a technique based on approximation of multi-variable functions by plane waves. The following theorem is a corollary of a result by Vostrecov and Kreines, 1961.

Theorem 10 *Let W_n is the set of weight vectors of dimension n and $\mathcal{N}(H_n)$ is the set of null points of homogeneous polynomial H_n of n variables. If $W_n \not\subset \mathcal{N}(H_n)$ for any H_n, then there exist continuous functions φ_{ik} of single variable such that the set of functions $\Phi_k(x)$ of the form $\Phi_k(x) = \sum_{i=1}^{k} \varphi_{ik}(w_i \cdot x)$, where $w_i \in W_n$, is dense in the set of all continuous functions.*

Last generalization can be done to multiple outputs, i.e. to vector functions $F : \mathcal{C}^n \to \mathcal{C}^m$, through Cartesian product of one dimensional output spaces. Thus we obtain general networks with multiple inputs and outputs.

5. Learning possibilities

There are various possibilities of learning algorithms for the chain architectures. We just briefly mention an application of three methods here, details can be found in Neruda and Štědrý, 1996.

First, it is possible to derive a variant of the back propagation learning algorithm (Rumelhart et al, 1986) for this architecture. The error propagates from top of the network to the bottom as usual, but it is important that the computation in the hidden layer follows the lateral connections of units. It means that the forward phase goes from left to right (considering the orientation in figure 10), while the backward computation proceeds against the direction of horizontal connections.

Imagine an incremental learning algorithm that gradually adds new hidden units placing each one as the leftmost unit in the hidden layer. After adding the unit we keep the old parameters frozen and adapt only the parameters of the new unit. This corresponds to adding a new term to the partial continued fraction while the old terms remain the same. If the adaptation algorithm is reliable[1] this sequence of partial fractions converges to the desirable function. In any case we can rely on some gradient adaptation algorithm with known problems of local minima. Alternatively, the Cascor algorithm can be used since the chain networks create a subclass of cascade networks.

[1]For instance, suppose quite unrealistic case that the function has a form of a power series. In this case we can use the Euler formula to explicitly compute parameters in each step.

An interesting approach uses a result due to Viskovatov (cf. Danilov et al, 1961) about the continued fraction expansion of rational functions. According to the following formula we can incrementally create a continued fraction based on the coefficients of the rational function:

$$\frac{a_{10} + a_{11}x + a_{12}x^2 + \ldots + a_{1n}x^n + \ldots}{a_{00} + a_{01}x + a_{02}x^2 + \ldots + a_{0n}x^n + \ldots} = \frac{a_{10}}{a_{00} + \frac{a_{20}x}{a_{10} + \frac{a_{30}x}{a_{20} + \ldots}}} \tag{31}$$

where

$$a_{mn} = a_{m-1,0}a_{m-2,n+1} - a_{m-2,0}a_{m-1,n+1}. \tag{32}$$

Having the training set, we can imagine a gradient learning algorithm that fits the data by a rational function. Using the above formula 31 one can directly set the parameters of a chain network. Moreover, it is not necessary to have this interlink, since we can directly derive an adaptive algorithm for the network parameters, which is based on the recursive formula 32. So far, this approach works only for one-dimensional case but it would be possible to use the theorem 10 to extend it to a multiple input dimension.

6. Discussion

In our considerations we used the simplest possible activation function of a single unit with respect to the form of continued fraction. It is known from the continued fraction theory, that convergence can be speeded up by the choice of polynomials of higher degree. The concrete usage of such transformation is a question of balance between the complexity of one neuron computations and the number of neurons in the network.

Approximation of functions by perceptron or RBF units can be seen as geometrically intuitive. This is not true in the case of our chain networks, where the activation function, especially in the complex domain, together with the gradual composition does not provide a simple view. On the other hand, it follows from the continued fraction properties that the obtained approximation should be optimal in a sense of the minimization of the number of multiplication and division operations.

Acknowledgments: This work was partially supported by GACR under grant 201/95/0976.

References

[1] Barron, A.R. (1993) Universal approximation bounds for superpositions of sigmoidal function, *IEEE Transactions on information theory*, **39**, 3.

[2] Danilov V.L. et al (1961) *Mathematical analysis*, Moscow, (in Russian).

[3] Euler L. (1739) De fractionibus continuis observatione, *Comm. Acad. Sci. Imper.*, Petropol, **11**.

[4] Fahlman S., Lebiere C. (1991) The cascade correlation learning architecture, In Touretzky D. (ed.) *Advances in Neural Information Processing Systems*, **2**, Morgan–Kaufman.

[5] Hirose, A., (1992a) Dynamics of Fully Complex-Valued Neural Networks, *Electronics Letters*, **16**, 1492–1493.

[6] Hirose, A. (1992b) Continuous Complex-Valued Back-Propagation Learning, *Electronics Letters*, **20**, 1854–1855.

[7] Neruda, R. Štědrý, A. (1995) Approximation Capabilities of Chain Architectures, *Proceedings of the ICANN'95*, EC2 & Cie: Paris, I. 575–580.

[8] Neruda, R. Štědrý, A. (1996) Learning in Cascade Neural Networks, Technical report, Institute of Comp. Sci., Prague.

[9] Rumelhart, D.E., Hinton, G.E., Williams, R.J., (1986) Learning Representations by Back–Propagating Errors. *Nature*, **323**, 533–536.

[10] Vostrecov B. A., Krejnes M. A. (1961) On approximation of continues functions by superpositions of plane waves, *Dokl. Akad. Nauk. USSR.*, **140**, 6, 1237–1240, Moscow.

[11] Wall H.S. (1948) *Analytic Theory of Continued Fractions*, Van Nostrand, New York.

Biquadratic Error Functions for the BP-networks

doktorand:
Mgr. Milan Rydvan
ÚI AV ČR, Pod Vodárenskou věží 2, 182 07 Praha 8
rydvan@ss1000.ms.mff.cuni.cz

školitel:
RNDr. Ladislav Andrej, CSc.
ÚI AV ČR, Pod Vodárenskou věží 2, 182 07 Praha 8
andre@uivt.cas.cz

obor studia:
Teoretická informatika

Abstrakt

Multilayer feed-forward neural networks, trained by the Back-Propagation algorithm can theoretically use any differentiable error function. Practically, however, almost only the summed square error function is used. That means the error function $\sum (y-d)^2$, summed over all training patterns and over all outputs, where d is desired output and y is actual output. This error function, however, depends only on the difference $|y-d|$, which can in some cases prove unsuitable.

This article studies an natural generalization of the standard error function, which does not suffer from this problem — a function of the type $E = \sum BIQ\,(d,y)$, where BIQ is a biquadratic function. The features and convergence of the networks, trained using this type of error functions is examined, and a possible method of the choice of its actual paramters is proposed. The method is then experimentaly tested on the stock prediction problem.

1. Introduction

This article is proposing a possible improvements in the usage of the multi-layer model of neural networks and its Back-Propagation training algorithm. We assume, that the reader is familiar with both the model and the algorithm - they can be found either in the original [2] or in most of the literature dealing with neural networks.

The BP-training algorithm of multi-layer neural networks is a gradient-descent method of minimizing an error function. This error function can generally be any differentiable function of the desired output d and the actual output y. Practically, however, the almost only error function used is summed square error function, proposed already by the authors of the BP-training algorithm in [2]. It is the function

$$E = \frac{1}{2}\sum_{t=1}^{p}\sum_{j=1}^{m}\left(y_{tj}-d_{tj}\right)^2 , \tag{33}$$

where p is number of the training patterns, m is number of the output neurons, y_{tj} is the actual and d_{tj} the desired output of the j-th output neuron on the t-th training pattern. This is a very general error function — it can be used for all kinds of numeric outputs. It is also a very natural one — the greater is the difference between the desired and the actual output, the greater is the error and the quadratic dependence strengthens the importance of major errors. It has, however, a property, that we do not always like. The error depends only on the difference between the desired and the actual output. That means, that every two pairs with the same difference between the desired and the actual output will have the same value of error.

An example of a case, when this feature might be unsuitable is our sample task — prediction of the stock prices. Our goal is to develop a prediction system, that would maximize our profit, if we abide by its predictions. Let us imagine we have two cases with the same difference between the desired and the actual value. The first case is that a strong rise of a stock was predicted, and in fact stagnation (no change) appeared. The second case is quite opposite — stagnation was predicted and a strong rise appeared instead. The difference between the predicted and actual value is the same in both cases, and yet the second case is much worse for us than the first one, the worse the greater the rise was. In the first case we have bought some shares (they were supposed to rise), but even though their price did not rise, we can easily sell them back for the same price as we bought them and thus lose only the transaction costs. In the other case we did not lose anything on the first glance — we did not trade, because the prediction said there would be no change of price. Looking more thoroughly, however, we find out, that we have lost a great chance for a profit. If we had bought the shares, we could have sold them for higher price and have a big profit. Thus we suffered an *opportunity loss*, larger than would be the actual loss in the first case. The error value assigned to the second case should be higher than that assigned to the first case.

This kind of problems will probably appear in more problems, that we try to solve using neural networks. We propose a partial solution — using a more general error function. In the following text we will describe one type of such functions - a biquadratic error function.

2. Neural Predictions

Neural networks are widely employed in predicting various phenomena. In financial predictions they proved to be a very useful tool and various methods how to improve their performance were developed. Many ways of data preprocessing, making them more suitable for the network, can be found in [4]. A very strong preprocessing method is Principal Component Analysis ([3]), that enables to reduce redundancy and ortonormalize the training data in the input space. Another methods address the problem of generalization — some of the approaches are forcing of a uniform internal representation (the hidden neurons' output is forced to have only certain values, see [5] for details) or adding a random noise into the training data, thus making the network to respond similarly on similar inputs ([7]).

The approach described in this article tries to improve the behaviour of the BP-networks in another way. It can be actually combined with the above methods of data preprocessing and generalization improvement, because it tries to improve another part of the BP-training algorithm — the error function. The problems with various importance of the training patterns were already faced in past. One of the solutions can be found for example in [1]. The idea was to have a variable learning rate, instead of a fixed one, different for various training patterns. That means, that the patterns, which are considered crucial for learning of the network, are presented to the network with a higher learning rate than those considered less important. The other approaches are similar — it is possible instead of increasing the learning rate for the important patterns to present them more often to the network, or even to discard some of the less important data from the training set.

There is a common idea behind these methods — we have (empirically) chosen some of the training data as more important and we try to make the network to learn them better. We do not change the error function — we only make longer (or more) steps through the same one. In some cases this can work very well. For example if our task was to make an alarm system for a nuclear power station, and a vast majority of the data we had available were from the quiet periods when nothing interesting happened, we might wish to stregthen the few data we had from the accidents or another interesting periods.

In our stock price example, however, the situation is different. It is difficult to decide, whether a training pattern, consisting of an input and a desired output, is important or not. On the other hand we can say, given the desired output and *the actual output*, how good or bad such a situation would be for us. In another words, how big error value should be assigned to it — but this error is not the same for every pair of patterns with the same difference between the desired and the actual output. This is impossible to achieve using the standard error function (33). We need a more general error function, that would be able to teach the network the extra knowledge we have.

3. Error Function

Biquadratic Error Function: We are now in the following situation: we choose a set of points in the two-dimensional space of the desired and actual outputs, for which we empirically decide what error value they should have. In the stock prediction example we can for instance make a table similar to Table 1.

	Actual output		
Desired output	0.25	0.50	0.75
0.25	0.5	2	4
0.50	1	0.5	1
0.75	4	2	0.5

Tabulka 1: Empirical error values

We predict stock price changes. The actual values are those predicted by the network, the desired ones are those really achieved on the market. All the price changes were transformed into interval $(0, 1)$ so that they could be predicted by neuron's output - 0 represents maximum fall of price, 1 represents maximum rise. Obviously the error is lowest in the case that the prediction was correct (the same actual and desired value) and highest when the prediction was completely wrong. The rest of the cases is interesting — they represent our extra knowledge. The cases where a price change occured, but the network predicted stagnation are worse for us, and thus are assigned higher error value, than those where a stagnation occured, while the network predicted a change.

This was an example — in the general case we will have a set of points $P = \{(d_i, y_i, E_i)\}$, representing the empirically chosen values of error for the given pairs of the desired and actual output. These are, however, discrete values and for the BP-training algorithm we need a differentiable error function. Generally we may approximate them by any differentiable function. In our experiments we have chosen a biquadratic function of the form

$$BIQ(d, y) = A_2 d^2 + A_1 d + A_2 y^2 + A_1 y + Cdy + D, \tag{34}$$

where $A_2, \ldots, D$ are constants. The reason for this choice is that it is a function which is very simple to compute, and easily differentiable — it is a multinominal in both its variables. It is a natural generalization of the standard error function (33), that allows the error value to be independent on the difference $|y - d|$. The actual values of the constants $A_2, \ldots, D$ can be obtained by any interpolation method, that will fit a biquadratic function through the points from P — we have chosen the least square method.

The error function then will be

$$E = \frac{1}{2} \sum_{t=1}^{p} \sum_{j=1}^{m} BIQ(d_{tj}, y_{tj}), \tag{35}$$

using the same notation as in (33).

Training algorithm: Now let us discuss how a change of the error function will affect the BP-training algorithm . From the formal point of view there are very few changes — the derivation remains almost the same. The gradient descent method is based on the equation

$$w_i := w_i - \alpha \frac{\partial E}{\partial w}, \tag{36}$$

where E is the error function, α is the learning rate and w is the weight we update. We apply this rule on every weight of every neuron, each is changed in the direction of decreasing error. We can express the crucial part $\frac{\partial E}{\partial w}$ by applying the chain rule of derivatives and we obtain

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial y} \cdot \frac{\partial y}{\partial w}, \tag{37}$$

where y is output of the updated neuron.

Only the first factor of this term depends on E — the rest will remain unchanged. In the case of a hidden neuron this first factor is computed using the already computed error terms propagated from the above layer (the basic principle of back-propagation) — this can again stay unchanged. Thus, the only part of the derivation that actually depends on the shape of the error function is the term $\frac{\partial E}{\partial y}$ in the case of an output neuron. This term is also easy to differentiate, because E is a multinominal in y.

Thus, we could see that formally the change of error function is very easy and actually does not affect the training algorithm much. In the following section we will however see that it affected the actual behaviour of the network very much, and in the way we desired.

4. Experimental Results

Sample problem: We have chosen the problem of stock price prediction. The goal was to predict the price change of the stock of the Czech Energetic Company ČEZ on the Prague Stock Exchange from the knowledge of five previous price changes, and the volume of trade, supply/demand relation and long-term position of the price on the previous working day. The input data were preprocessed by transforming into interval $(0, 1)$ and by applying Principal Component Analysis ([3]). The details about the preprocessing are not important, because they were identical for all the error functions; they can be found in [6] — the same methods like there were used. All the networks had the same architecture (9-15-1) and learning rate (0.01). We carried out 100 experiments. In order to minimize the random influences, we generated the initial (random) values of weights and thresholds only once during each experiment and used these common values for both types of network. The available data were randomly divided into the training set (about 90% of the data), used for the training of the network and the test set, that is not presented to the network during the training process. It was used for evaluating generalization abilities of the networks. This division into training and test set was also identical for both networks trained in one experiment. Thus during each experiment the two trained networks differed only in the error function — we trained a network for each of the functions (33) and (35), the biquadratic function was obtained by interpolating the points in Table 1 by the least square method. The networks were trained until they reached an error goal, using the patterns from the training set. An error goal showing the best results was chosen experimentally for each error function.

Measures of success: We have chosen four different ways to measure how successful networks were in their predictions. In the results below each of these measures is given as the average over the 100 experiments and stated apart for the training set and the test set. The first measure is simply a mean error over all the patterns from the particular set. The second and third measure are ratios of the predictions, that were in the same category as the result actually achieved on the stock exchange. In the case of *3-way correctness* the categories are rise (price change $\geq 0.5\%$), stagnation (price change $\in (-0.5\%, 0.5\%)$) and fall (price change $\leq -0.5\%$). In the case of *direction correctness* we have only two categories — rise (price change ≥ 0) and fall (price change ≤ 0). These measures, however, sometimes show results that are strange. For example a network that would always predict utter stagnation (0 price change) would be 100% successful measured by direction correctness, but we would definitely not call such a network successful in the natural meaning of the word. The other two measures can suffer similar kind of problems, though not so extreme. That is why the last of the measures intends to model the profit that a real broker would obtain if he abided by the predictions of our network. Let us denote the predicted price change y, the one really achieved d and the transaction costs (we assume them to be proportional to the value of traded stock) c, all in per cents. We assume that the broker has a certain amount of shares of the given stock. If $y \leq -c$, he will sell these shares and buy them back next day, when (according to the prediction) the price should be lower. Analogically, if $y \geq c$, he will buy once more that amount and sell it back next day. In another case he will do nothing — the possible profit would not cover the transaction costs. The profit p from a single trading day then can be defined as follows:

- $y \leq -c \Rightarrow p = -d - c$ ... we sold and bought back
- $y \in (-c, c) \Rightarrow p = 0$ ... we did not do anything
- $y \geq c \Rightarrow p = d - c$ ... we bought and sold back

The parameter c is set to 0.2% — the value of transaction costs on the Prague Stock Exchange.

Results: The columns of Table 2 show the following information: the first two show the error function used and whether the results were achieved on the training set or test set. Columns 3-6 show the average results of the networks achieved measured by the measures stated above. The column labeled 'Trading' state the ratio of the *trading predictions*, that means predictions that lead to a buying or selling of the stock, from all the cases. The columns labeled 'Succ.' and 'Crit.err' are ratios of the trading predictions only. 'Succ.' shows the ratio of successful predictions, that means those where the price did actually rise or fall as the prediction said and we gained, 'Crit.err.' analogically shows the ratio of critical errors, when the price went the opposite way and we suffered a loss. The last column then states the number of training cycles, needed to train the networks. Certainly more important for us are the results achieved on the test set, that show us how well the network behave on inputs it did not know during training. The training results are interesting when compared to the test ones, indicating, how well a given network generalizes.

E.f.	Set	Mean err.	3-way cor. [%]	Dir.cor. [%]	Profit [%]	Trading [%]	Succ. [%]	Crit.err. [%]	Cyc.
(33)	Train	0.1419	60.44	81.72	0.4732	71.32	64.32	11.82	767
	Test	0.1746	49.76	72.45	0.1110	72.23	50.38	24.12	
(35)	Train	0.2083	44.81	78.47	0.3712	91.47	55.43	18.21	179
	Test	0.2349	39.51	73.35	0.2035	92.60	49.04	23.11	

Tabulka 2: Results of the experiments

There is a number of phenomena that deserve a comment in these results. First, the achieved profits are not bad. The daily profits of 0.11% and 0.2% correspond to cca 30% to 50% per annum, assuming 250 trading days per year. These are netto values, the stock exchange fees are already comprehended, achieved in a generally decreasing market environment (the average price change was about -0.075%). Even though we must take into account that this is only a model of profit (it does not take into account all factors, for example that the shares are sometimes not available to buy/sell), it is good for making a rough insight how high the profit might be.

We can see (and it is confirmed by examination of the original, non-averaged data) that the network with the best gains usually have a very high direction correctness, but their mean error and three-ways correcntess are usually poor. This is because the biquadratic error function is doing its job — by increasing the error value of the mistakes made when predicting small or none price changes it makes the network to be more courageous. If we abide by its prediction, we would trade in more than 90% of cases, even though in the whole data there is only 73% of cases, where the trade makes profit. That necessarily leads to increasing of the error rate. Surprisingly, the success rates decreased and critical error rates increased only on the training data, but on the test data they remained almost the same. This, together with the fact that a more courageous network uses some more trading chances, that a less couragous one would not detect, leads to a fact that while on the training set the profits of (35) were lower than those of (33), on the test set it is opposite. That can mean that the generalization abilities of (35) are higher, that the networks trained using this function discovered some deeper dependencies in the training data.

Another interesting result is that the networks trained using (35) were also much faster to train — they needed only about one fourth of the cycles to train, compared to (33). The last remark: because of the rather large number of experiments carried out, the results are statistically reliable. All the watched quantities were subjected to pair t-test for the hypothesis that the mean values of the random quantities for error functions (33) and (35) are equal. All of these hypotheses were rejected with the significance level (probability, that we reject a valid hypothesis) lower than 1%.

5. Conclusions

In this work new forms of error function for multi-layer neural networks trained by the BP-training algorithm were proposed. The goal was to enable inclusion of the empirical information we have about the training patterns, which have the same difference between the desired and the actual output. This was impossible using the standard error function, based just on this difference.

We have seen, that achieving this goal is possible, that training of the networks using this forms of error functions converges to reasonable results. Measured by the profit-based model of success, the networks trained using the new error functions achieved even better results than those trained using the standard one.

The aim of this article was to examine, whether the neural networks' performance in predictions could be improved by using a more general error functions and it seems it could. There certainly remain many questions worth further research, besides others to examine another forms of error functions and another measures of success. It might be interesting to derive an error function based on the profit model — to try not to minimize error, but to maximize profit. The rather good results of the networks trained using the profit-based biquadratic function are promising.

References

[1] Casimir C. Klimasauskas, "Neural Network Techniques" in G. J. Deboeck, "Trading on the Edge", John Wiley & Sons, 1994 pp. 3-26

[2] D. E. Rumelhart, G. E. Hinton, R. J. Williams, "Learning representations by backpropagating errors." Nature, 1986

[3] T. Masters, "Advanced Algorithms for Neural Networks." John Wiley & Sons, 1995

[4] T. Masters, "Neural, Novel & Hybrid Algorithms for Time Series Prediction." John Wiley & Sons, 1995

[5] I. Mrázová, "Internal Representation in BP-Networks. " PhD thesis, ICS CAS, 1997

[6] M. Rydvan, "System behaviour predicting using multi-layer neural networks." Diploma thesis, MFF UK Prague 1997. Postscript version can be found at `http://www.ms.mff.cuni.cz/~rydvan/diploma.ps`

[7] J. Sietsma, R. J. F. Dow, "Creating Artificial Neural Networks That Generalize." Neural Networks, Vol. 4, 1991, pp. 67-79

Metody podpory rozhodování v medicíně

doktorand:
ING. PETR HANZLÍČEK
EuroMISE Centrum UK a AV ČR, Pod vodárenskou věží 2, 180 00, Praha 8
hanzlicek@euromise.cz

školitel:
DOC. RNDR. JANA ZVÁROVÁ, CSC.
EuroMISE Centrum UK a AV ČR, Pod vodárenskou věží 2, 180 00, Praha 8
zvarova@euromise.cz

obor studia:
Teoretická informatika

Metodám podpory rozhodování v medicíně věnovalo velkou pozornost mnoho významných odborníků. Například Braude [1] pojímal medicínskou informatiku jako aplikaci počítačové vědy do lékařství a zdravotnictví, která zahrnuje zejména následující oblasti:

- reprezentace znalostí, principy vytváření bází znalostí a databází v medicíně
- studium vlastností přirozeného jazyka v medicínském kontextu a metody zpracování textové informace
- metody umělé inteligence, aplikované na báze znalostí a při vytváření expertních systémů
- studium lidských faktorů při ukládání a využívání znalostí v počítači a při používání konzultačních systémů
- metody rozhodování a řešení medicínských problémů
- analýza podstaty medicínských znalostí a jejich významu pro klinické aktivity
- studium vlivu pokročilých počítačových technologií na ustálené postupy při zjišťování, přenášení a využívání medicínských znalostí

Zdůraznění velkého významu medicínské informatiky pro podporu rozhodování, zejména v lékařem řízené péči o nemocného, nepochybně přispělo k jejímu rozvoji, zejména v USA.

V medicíně je rozhodování založeno na dvou základních typech znalostí: vědeckých znalostech (poznání podstaty problémů a procesů biomedicínským výzkumem) a empirických znalostech (zkušenostech, získaných při diagnostice a léčbě pacientů). Oba typy znalostí jsou popisovány v učebnicích a dalších odborných publikacích a zejména vědecké znalosti jsou vyučovány na lékařských fakultách. Vědecké znalosti (know-how) jsou kognitivního typu, tj. poznáváme podstatu biologických procesů, vztahy mezi patofyziologickými podmínkami a příznaky nemocí. Klinická zkušenost, soustředěná v dobře vedené dokumentaci o nemocných a v počítačové formě uložená v databázi nemocných, tvoří empirické znalosti (know-why): lékař zjistí určité příznaky a na jejich základě rozpozná chorobu pacienta.

Na rozdíl od činnosti lidského mozku, proces rozhodování za pomoci počítače musí být zcela přesně popsán. Systémy pro podporu rozhodování mohou tedy pracovat podle zvoleného modelu rozhodování na základě různých metod. Tyto metody lze zhruba rozdělit na statistické, logické a heuristické. Ke statistickým metodám patří například známé Bayesovské rozhodování, mezi logické metody rozhodovací stromy a k heuristickým metodám řadíme takové expertní systémy, kdy odvozování závěrů o vyšetřovaném pacientovi probíhá s využitím bází medicínských dat a znalostí pomocí elementárních rozhodnutí typu ANO-NE.

S rozvojem nových informačních technologií se zdá být přirozené, že v blízké budoucnosti se počítače stanou cenným pomocníkem lékaře v náročných rozhodovacích situacích. To předpokládali již v šedesátých letech Ledly a Lusted [2], ale jejich vize se v tehdy nenaplnila. Vyvíjené systémy pro podporu rozhodování byly příliš ambiciózní. Snažily se pokrýt problémy z celé klinické medicíny a málo se orientovaly na praktické potřeby lékařů, kde by mohly poskytnout účinnou pomoc i při poměrně málo rozvinutém technickém vybavení počítačů. Toto počáteční selhání systémů pro podporu rozhodování 60. let vedlo k tomu, že v 70. letech se pozornost soustředila na vytváření systémů, zaměřených do užších oblastí medicíny. Typickým příkladem je systém pro diagnostiku náhlých bolestí břicha vyvinutý v Leedsu [3] a diagnostiku žloutenky, jejichž báze dat se týkají úzké medicínské oblasti a k odvozování závěrů využívají Bayesovský model rozhodování. Současně se začaly vyvíjet techniky umělé inteligence podporující lepší formalizaci medicínských znalostí. K významným systémum založeným na znalostech patří například PIP [4], CASNET [5], MYCIN [6] a INTERNIST-1 [7], které používaly různé techniky pro reprezentaci a ukládání znalostí [8].

Systémy pro podporu rozhodování jsou v současné době předmětem intenzivního interdisciplinárního výzkumu. Předmětem mého příspěvku budou výsledky rešerše, provedené prostřednictvím MEDLINE a na základě informačních zdrojů z kongresů MEDINFO, MIE, z řady Yearbook of Medical Informatics a dalších literárních zdrojů.

Podpora rozhodování je často založena na znacích o nemocných, ukládaných v rozsáhlých medicínských databázích. V tomto případě je cílem výzkumu vyhledávat takové znaky, které umožňují kvalitní rozhodování pro daný medicínský problém. Pro tento účel jsou vyvíjeny různé metody pro redukci a konstituci dat a výběr relevantních znaků pro rozhodování.

Přístup, kterým se podrobněji zabývám, vychází z metod teorie informace, podrobně popsaných v pracích [9], [10], [11]. Přístup je založen na informačních mírách stochastické závislosti a podmíněné stochastické závislosti. Jedná se o nezáporné číselné charakteristiky, vyjadřující sílu stochastické závislosti mezi dvěma proměnnými (resp. sílu podmíněné závislosti mezi dvěma proměnnými při zadané hodnotě třetí proměnné). Základní požadavek, který informační míra musí splňovat, je nulovost právě tehdy, jsou-li uvažované dvě proměnné nezávislé (nebo podmíněně nezávislé). Tyto míry jsou studovány v teorii informace jako nástroje pro odhad Bayesovského rizika. Důležité vlastnosti měr závislosti byly zmíněny již v 60. letech jedním ze zakladatelů české školy teorie informace A. Perezem [12][13]. Nejčastěji používanou mírou závislosti je míra, založená na klasické Shannonově informaci. Kromě této míry existují další, založené na obecném konceptu f-informace, navrženém I.Vajdou [14][15]. Dále bylo ukázáno, že multiinformace, sloužící jako míra vzájemné závislosti [17][18], má blízký vztah k podmíněné Shannonově vzájemné informaci, která slouží jako míra podmíněné stochastické závislosti.

Připomeňme definice některých informačních měr pro diskrétní případ. Nechť V je neprázdná konečná množina proměnných, pro každou proměnnou $v \in V$ symbol $\mathbf{R}_v$ označuje konečnou neprázdnou množinu jejích možných hodnot. Pro každé $\emptyset \neq A \subset V$ budeme označovat $\mathbf{R}_A$ kartézský součin $\prod_{v \in A} \mathbf{R}_v$.

Nechť P je pravděpodobnostní distribuce na $\mathbf{R}_V$. Máme-li $\emptyset \neq A \subset V$, marginální distribuce P na $\mathbf{R}_A$ označovaná P^A je definována vzorcem

$$P^A(y) = \sum \{P(y,z);\, z \in \mathbf{R}_{V \setminus A}\} \qquad \text{pro každé } y \in \mathbf{R}_A.$$

Výraz P^V budeme zkracovat na P a výraz P^i bude zkrácenou formou $P^{\{i\}}$ kde $i \in V$. Protože $\sum\{P(z);\, z \in \mathbf{R}_{V\setminus\emptyset}\} = \sum\{P(z);\, z \in \mathbf{R}_V\} = 1$ podle definice distribuce pravděpodobnosti, přijmeme v následujících vzorcích konvenci $P^{\emptyset}(-) = 1$.

Je-li $\emptyset \neq A \subset V$, *entropie* $H(A)$ je definována vzorcem

$$H(A) \;=\; \sum\{\, P^A(x) \cdot \ln \frac{1}{P^A(x)}\,;\, x \in \mathbf{R}_A \;,\; P^A(x) > 0\,\}\,,$$

s tím, že $H(\emptyset) = 0$. *Multiinformace* $M(A)$ je v tomto případě definována vzorcem

$$M(A) \;=\; \sum\{\, P^A([x_v]_{v\in A}) \cdot \ln \frac{P^A([x_v]_{v\in A})}{\prod_{i\in A} P^i(x_i)}\,; \quad \text{kde } x_v \in \mathbf{R}_v \;\forall\, v \in A \;,\; P^A([x_v]_{v\in A}) > 0\,\},$$

s tím, že $M(\emptyset) = 0$. Obě tyto funkce jsou na V nezáporné.

Pro každou dvojici množin $A, B \subset V$, $A \cap B = \emptyset$ *Shannonova vzájemná informace* je relativní entropie $P^{A\cup B}$ vzhledem k součinu P^A a P^B:

$$I(A;B) \;=\; \sum\{\, P^{A\cup B}(a,b) \cdot \ln \frac{P^{A\cup B}(a,b)}{P^A(a) \cdot P^B(b)}\,;\, a \in \mathbf{R}_A \;,\; b \in \mathbf{R}_B \;,\; P^{A\cup B}(a,b) > 0\,\}.$$

Vzhledem k tomu, že (viz. [10]) $0 \leq I(A;B) \leq \min\{H(A), H(B)\}$ je *Shannonova informační míra* A na B, která je definována jako $\delta(A|B) \;=\; I(A;B)/H(A)$ vždy reálné číslo mezi 0 a 1.

Pro každou trojici množin $A, B, C \subset V$, $A \cap B \cap C = \emptyset$ je *Shannonova podmíněná vzájemná informace* definována jako

$$I_c(A;B|C) \;=\; \sum\{\, P^{A\cup B\cup C}(a,b,c) \cdot \ln \frac{P^{A\cup B\cup C}(a,b,c) \cdot P^C(c)}{P^{A\cup C}(a,c) \cdot P^{B\cup C}(b,c)}\,; \quad a \in \mathbf{R}_A \;,\; b \in \mathbf{R}_B \;,\; c \in \mathbf{R}_C \;,\; P^{A\cup B\cup C}(a,b,c) > 0\,\}.$$

Shannonovu podmíněnou vzájemnou informaci lze rovněž spočítat pomocí multiinformace

$$I_c(A;B|C) \;=\; M(A \cup B \cup C) + M(C) - M(A \cup C) - M(B \cup C).$$

References

[1] Braude R.M.: Environmental and person factors in secondary career choice of graduates of medical informatics training programs. Disertation. University of Nebrasca, Lincoln Nebrasca 1987.

[2] Ledley R.S., Lusted L.B. : The use of electronic computers in medical data processing aids in diagnosis, current information retrieval and medical record, IRE Trans. Med. Electron. ME-7, 31-47,1960

[3] Knill-Jones J.R., Stern R.B., Girmes D.H., Maxwell R.P., Thompson P.H., Williams R. : use of sequential Bayesian model in diagnosis of jaundice by computer. British Medical Journal 1. 1973, 530-553

[4] Pauker S.G., Gorry G.A., Kassirer J.P., Shwartz W.B. : Towards the simulation of clinical cognition: Taking a present illness by computer. The Americal Journal of Medicine 60, 1976, 981-995

[5] Weiss M.C., Kulikowsky C.A., Safir A.K. : Glaucoma consultation by computers. Computers in Biology and Medicine 1, 1978, 25-40

[6] Shorliffe E.H. : Computer based medical consultation: MYCIN, North Holland, Amsterdam 1976

[7] Miller R.A., Pople H.E., Myers J.D. : Internist-1 an experimental computer based diagnostic for general internal medicine. New england Journal of Medicine, 307, 1982, 468-476

[8] Hucklenbroich P. : Methodological principles in medical knowledge programming. Part I. Artificial Intelligence in Medicine 3,1991, 113-123

[9] Zvárová J., Studený M., Preiss J. : On extracting relevant information from medical data. MEDINFO 96 proceedings, J. Brender et al., Amsterdam : IOS Press, 1996, 649-653

[10] Zvárová J., Studený M. : Information theoretical approach to constitution and reduction of medical data. Int J Med Inf 1997 Jun;45(1-2), 65-74

[11] Zvárová J., Tomečková M., Štefek M., Boudík F., Zvára K. : Decision support and data analysis tools for risk assessment in primary preventive study of atherosclerosis. MEDINFO 97 proceedings, C. Pappas et al., Amsterdam : IOS Press, 1997, 625-628

[12] Perez A. : Information-theoretical risk estimates in statistical decision, Kybernetika 3, 1967, 11-21

[13] Perez A. : ε-admissible simplifications of the dependence structure of a set of random variables, Kybernetika 13, 1977, 439-449

[14] Vajda I. : On the f-divergence and singularity of probability measures. Periodica Math Hung 2, 1972, 223-234

[15] Vajda I. : Theory of statistical inference and information, Kluwer, Dordrecht, 1989

[16] Studený M. : Asymptotic behaviour of empirical multiinformation, Kybernetika 23, 1987, 124-135

[17] Studený M. : Multiinformation and the problem of characterization of conditional independence relations, Probl. control info. theory 18, 1989, 3-16

[18] Zvárová J. : Informační míry statistické závislosti a výběrové vlastnosti zobecněné entropie řádu α, thesis, Institute of information theory nad automation and Charles University, Praha, 1973

[19] Zvárová J. : On measures of statistical dependence, Časopis pro pěstování matematiky 99, 1974, 15-29

[20] Zvárová J., Perez A., Nikl J., Jiroušek R. : Data reduction in computer-aided medical decision making, van Bemmel J.H., Ball M., Wigertz O., Proceedings MEDINFO83, North Holland, Amsterdam, 1983, 450-453

[21] Zvárová J., Srb B., Malý Z., Martan A. : Computer supported information analysis of cardiotocographic parameters, Willems J.L, van Bemmel J.H., Michel J., Progress in computer- assisted function analysis, North Holland, Amsterdam, 1988, 339-344

[22] Ledley R.S. : Computer aids to medical diagnosis. J. Amer. med. Assoc. 196, 1966, 933

[23] Bemmel van J.H., Zvárová J. : Medical decision making: diagnostic strategies and expert systems, North Holland, Amsterdam 1991.

[24] Dombal de F.T.: Leaper D.J. et al. : Computer aided diagnosis of acute abdominal pain. Britsh Medical Journal 2, 1972, 9-13

[25] Adam I.D., Chan M., Clifford P.C. et al. : Computer adided diagnosis of acute abdominal pain: a multicentre study. British Medical Journal 293, 1986, 800-804

[26] Wyatt J. : Computer based knowledge systems. The Lancet 338,1991,1431-1436

[27] Sawar M.J., Brennan, T.G., Cole, A.J., Stewart. J. : An Expert System for PostOperative Care. On Proceedings MEDINFO92, North Holland, Amsterdam 1992

[28] Kahn M, Steib S., Fraser V, Dunagen W. : An expert system for culture-based infection surveillance In Proceedings of SCAMC-93

[29] Edwards G., Compton P., malor R., Srinivasan A., Lazarus L. : PEIRS a pathologist maintened expert system for the interpretation of chemical pathology reports. Pathology 1993,25,27-34

[30] Snow M.G., Fallat R.J., Tyler W.R., Hsu, S.P : Pulmonary Consult: Concept to application of an expert system. Journal of Clinical Engineering 13,3, 1988, pp.20+-20

[31] Dojat M., Sayeettat C. : A realistic model for temporal reasoning in real-tome patient monitoring. Applied Artificial Intelligence 2, 10,.1996

[32] Adlassnig, Horak W.: Hepaxpert I. : Automatic interpretation of tests for hepatitis A and B., MD Computing 8,2,1991,118-119

[33] Laires M.F., Laidera M.J., Christensen J.P. : Health in the new communications age, IOS Press, Amsterdam, Oxford, Tokyo, Washingon D.C. 1995

[34] Barahona P., Christensen J.P. : Knowledge and Decision in Health Telematics, IOS Press, Amsterdam, Oxford, Washington,DC, Tokyo 1994

[35] Nola J., Mc Nair P., Brender J. : Factors influencing the transferability of medical decision support systems. Int. J. of Biomedical Commputing 27, 1991, 7-26

[36] Jiroušek R. : Bayes. sítě - moderní technologie umělé inteligence. Lékař a technika 4/98,

[37] de Graaf PM, van den Eijkel GC, Vullings HJ, de Mol BA : A decision-driven design of a decision support system in anesthesia. Artif Intell Med 1997 Oct;11(2):141-153

[38] Rakus-Andersson E., Gerstenkorn T. : Application of a fuzzy model of making a decision to choose a medicine in the case of symptoms prevailing after the treatment. MEDINFO 97 proceedings, C. Pappas et al., Amsterdam : IOS Press, 1997, 277-279

[39] Smyth RL : Evidence-based pediatric pulmonary medicine: how can it help? Pediatr Pulmonol 1998 Feb;25(2):118-127

[40] Abott A. P., Quirolgico S., Manchand R., Canfield K., Adya M. : Can the US minimum data set be used for predicting admissions to acute care facilities? MEDINFO 98 proceedings, B. Cesnik et al., Amsterdam : IOS Press, 1998 IMIA, 1318-1321

[41] Kagolovsky Y., Mohr R. J. : A structured model for evaluation of information retrieval. MEDINFO 98 proceedings, B. Cesnik et al., Amsterdam : IOS Press, 1998 IMIA, 171-175