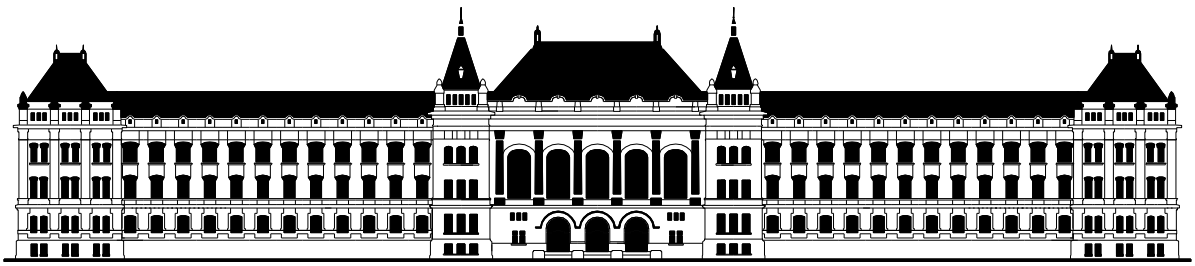
IEEE HUNGARY SECTION

CIRCUITS, SYSTEMS AND COMPUTERS JOINT CHAPTER

INSTRUMENTATION AND MEASUREMENT & ENGINEERING IN MEDICINE AND BIOLOGY JOINT CHAPTER

PROCEEDINGS OF THE 19TH PHD MINI-SYMPOSIUM

JANUARY 30, 2012.

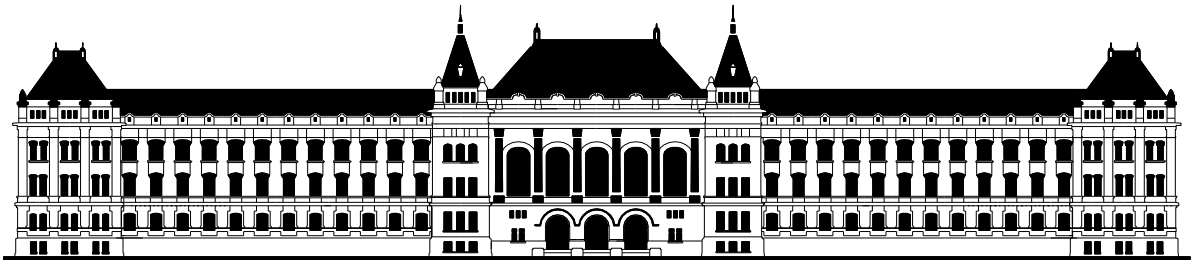


BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS

DEPARTMENT OF MEASUREMENT AND INFORMATION SYSTEMS

PROCEEDINGS
OF THE
19TH PHD MINI-SYMPOSIUM

JANUARY 30, 2012.
BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
BUILDING I



BUDAPEST UNIVERSITY OF TECHNOLOGY AND ECONOMICS
DEPARTMENT OF MEASUREMENT AND INFORMATION SYSTEMS

Conference Chairman:
Béla PATAKI

Organizers:
Péter Zoltán CSURCSIA
Tamás DEMIÁN
Róbert GALAMBOS
Péter MARX
Vilmos PÁLFI

Homepage of the Conference:
http://minisy.mit.bme.hu/

Sponsored by:
IEEE Hungary Section (technical sponsorship)
Schnell László Foundation

ISBN 978-963-313-046-9

FOREWORD

This proceedings is a collection of the lectures of the 19th PhD Mini-Symposium held at the Department of Measurement and Information Systems of the Budapest University of Technology and Economics. The main purpose of these symposiums is to give an opportunity to the PhD students of our department to present a summary of their work done in the preceding year. It is an interesting additional benefit, that the students get some experience: how to organize such events. Beyond this actual goal, it turned out that the proceedings of our symposiums give an interesting overview of the research and PhD education carried out in our department. The lectures reflect partly the scientific fields and work of the students, but we think that an insight into the research and development activity of the department is also given by these contributions. Traditionally our activity was focused on measurement and instrumentation. The area has slowly changed during the last few years. New areas mainly connected to embedded information systems, new aspects e.g. dependability and security are now in our scope of interest as well. Both theoretical and practical aspects are dealt with.

This is the first time that the proceedings will not be published in printed form, it has turned out that nowadays the web publication of symposium lectures is enough. This new form has some advantages, but it has some challenges as well. We hope that the advantages will dominate.

The papers of this proceedings are sorted into some main groups. These are Embedded and Intelligent Systems; Measurement and Signal Processing; Model-based Software Engineering and Knowledge Representation. The lectures are at different levels: some of them present the very first results of a research, others contain more new results. Some of the first year PhD students have been working on their fields only for half a year, therefore they submitted two-page papers. The second and third year students are more experienced and have more results; therefore they have four-page papers in the proceedings.

During this nineteen-year period there have been shorter or longer cooperation between our department and some universities, research institutes and firms. Some PhD research works gained a lot from these connections. In the last year the cooperation was especially fruitful with the Vrije Universiteit Brussel Dienst ELEC, Belgium; Thales Rail Signalling Solutions Ltd., Prolan Process Control Co., Nádor Systemshouse Ltd., OptXware Research & Development Ltd., IBM Data Storage Systems Ltd., IBM ISC Ltd., Innomed Medical Zrt., GAMAX kft., ImSoft kft, Robert Bosch kft., NI Hungary kft.

We hope that similarly to the previous years, also this PhD Mini-Symposium will be useful for the lecturers, for the audience and for all who read the proceedings.

Budapest, January 19, 2012.

Béla Pataki

Chairman of the PhD Mini-Symposium

LIST OF PARTICIPANTS

Participant	Advisor	Starting Year of PhD Course
BÁNYAI, Mihály	STRAUSZ, György	2009
CSERPÁN, Dorottya	HORVÁTH, Gábor	2011
CSURCSIA, Péter Zoltán	KOLLÁR, István	2010
DEMIÁN, Tamás	PATARICZA, András	2010
ENGEDY, István	HORVÁTH, Gábor	2009
EREDICS, Péter	DOBROWIECKI, Tadeusz	2009
GALAMBOS, Róbert	SUJBERT, László	2010
GÁTI, Kristóf	HORVÁTH, Gábor	2011
GYÖRKE, Péter	PATAKI, Béla	2011
HEGEDÜS, Ábel	VARRÓ, Dániel	2009
HORVÁTH, Áron	HORVÁTH, Gábor	2011
LACZKÓ, Péter	FEHÉR, Béla	2009
MARX, Péter	ANTAL, Péter	2010
OLÁH, János	MAJZIK, István	2009
ORBÁN, Gergely	HORVÁTH, Gábor	2009
PÁLFI, Vilmos	KOLLÁR, István	2010
SÁRKÖZY, Péter	ANTAL, Péter	2009
UJHELYI, Zoltán	VARRÓ, Dániel	2009
VÖRÖS, András	BARTHA, Tamás	2009

Program of The MINI-SYMPOSIUM

Conference Schedule

Time	January 30, 2012
8:30	Conference Opening Opening Speech: JOBBÁGY, Ákos
8:30	Embedded and Intelligent Systems
9:50	Cofee break
10:20	Model-based Software Technology and Knowledge Representation
11:40	Lunch break
13:00	Measurement, Signal Processing and Intelligent Systems
14:20	Cofee break
14:50	Intelligent Systems

EFFICIENT MULTITHREADED TASK SCHEDULER FOR VISUALIZATION AND DATA PROCESSING

Péter LACZKÓ
Advisors: Béla FEHÉR, István MIKLÓS

I. Introduction

Genomics is the science of whole-genome scale phenomena, with applications ranging from phylogenetic and evolutionary studies to disease research and pharmaceutics. The ever-increasing throughput and declining cost of next-generation sequencing (NGS) allows for studies with ever higher accuracy and wider scope. However, scaling up NGS experiments to the whole-genome level leads to enormous data sizes and poses a substantial challenge to traditional bioinformatics, setting the stage for methods that ease the handling and processing of such enormous data sets.

In our paper we start with an overview of our genomics research project and characterize the abstract informatics problem that can be generalized from the methods used in this particular study. We then briefly describe the visualization and data processing framework we developed as our solution. The focus of this paper is on the multithreaded task executor component of the framework; its design and performance analysis are presented in detail.

II. Genomic Rearrangement Breakpoint Detection

Extensive genomic rearrangements are observed in many cancer cells [1]. The accurate mapping of the tumor cell's rearranged genomic landscape allows for the discovery of oncogenes created by these structural variations as well as the classification of an unknown cancer case based on rearrangement event frequency. NGS techniques allow for such analysis with greater accuracy than traditional methods, giving rise, however, to a different range of challenges.

The output of next-generation sequencing consists of a large number of short (35-200 basepair long) reads originating from and covering manifold the genome of interest. These reads are mapped back to a reference genome supposedly similar to the one sequenced. There are two ways that these short reads can indicate the presence of possible breakpoints. First, if two halves of the same read map to distinct locations on the reference the read is likely to have covered a breakpoint in the original genome (split-read mapping). Second, if a pair of reads is known to have originated from within a known distance to each other yet they map significantly farther away on the reference, it is explainable by a rearrangement breakpoint between the two members (paired-end mapping).

Our combined representation of these two kinds of evidences was inspired greatly by [2]. In short, every such evidence supports a range of possible breakpoints. As a breakpoint may connect any two coordinates along the genome it can be represented as a point in a two-dimensional integer coordinate system. A mapped split-read or mate-pair supports only a limited set of breakpoints that are confined to the interior of a polygon in this coordinate system; this polygon is a square for split reads and a trapezoid for mate-pairs. The intersection of a number of these polygons represents the possible breakpoint that is supported by all the evidences involved.

The advantage of the above representation, besides being as general as possible, is that it is very intuitive in the sense that there is a clear correspondence between concepts of the problem domain (evidences, breakpoints, intersections) and their representation. Its downside, however, is that it is impossible to physically create such a graph on a basepair resolution for a whole-genome data set, as both axes of the coordinate system would extend to 3×10^9 in the case of the human genome. Therefore

a software tool that could visualize the evidence map of a real data set and would allow for the efficient execution of a wide variety of algorithms on it is of great value to the genomics researcher. Next, we briefly generalize this requirement and present our generic software library as a solution.

III. The Data Visualization and Processing Framework

A. *Requirements*

Consider a problem domain which can be represented in a very large (but finite) 1, 2 or 3 dimensional coordinate system. Given an input set of an arbitrarily large number of objects, our system must allow for visualization of any part of the chart, enumeration of the input data which was used for generating that part, and the execution of a custom algorithm on this data.

There is no restriction on the nature of the data objects as long as they satisfy a few (sometimes soft) conditions:

- They are independent, that is, any object is fully interpretable without any other objects.
- They can be assigned a one, two or three dimensional ,,bounding box" with integer coordinates that denotes their place in the coordinate system, which implies that
- they have a natural ordering.
- They are ,,small", that is, the size of the chart area that an individual object affects is a negligible portion of the whole coordinate system.

B. *Solution*

Our solution is a Java class library that exposes functionality to allow third party applications fulfill the above requirements; or, from a different approach, it encapsulates all domain independent functionality of a concrete visualization and data processing application for the breakpoint localization problem. See Figure 1 for a block diagram of the library integrated.

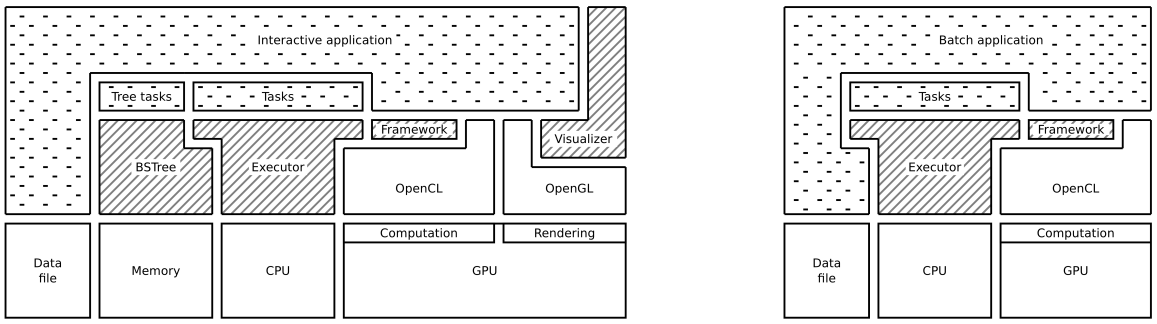


Figure 1: Functional blocks of the framework (dashed) interoperating with the custom (interactive or batch) application (dotted)

The main blocks of functionality are a binary spacing tree-based sorted cache of the current working set of objects (BSTree), the Executor module responsible for application task scheduling, and a visualizer engine which takes the burden of creating a usable interactive 3D-accelerated application off the integrator of our library. The application can submit general computational tasks on the collection of objects present in the working set which are ignorant on the underlying caching scheme; these tasks can also be executed in batch mode on the whole data set. In addition to these, specialized tasks aware of the underlying tree structure can also be executed.

The cache is based on a binary-, quad- or octree, depending on the dimensionality of the data (see the schematic illustration for a 2D quadtree in Figure 2 A). A tree node can be either nonleaf, with all possible children existing, or leaf, storing the data that belongs to the appropriate portion of the

entire coordinate system. The objects which overlap with this region are stored in leaf nodes, indexed separately by all coordinates. The indexing is implemented with the TreeSet collection of the Java language, which is effectively a binary search tree.

This double-layered design allows for fast but imprecise queries for a given region (in the BSTree), yet does not introduce much space overhead as the large number of objects are actually stored in a search tree (instead of a spacing tree). The multiple indexing of objects in each leaf node makes sorted retrieval of objects in any region possible, by first traversing all nodes that have an intersection with the region of interest and then merging the sorted object sets of these nodes (in linear time).

The fact that objects can be retrieved in order allows for a computational model that can handle object interdependecies. We assume that the density of input objects can be so high that an algorithm processing a region cannot load all its contents into memory; in other words, it has to access the data sequentally. Generally, this would practically inhibit the implementation of algorithms that do not process all objects independently. However, since the data will be sorted, the algorithm can keep track of a ,,sweeping pointer" across one axis of the entire coordinate system, and can assume that no objects with already passed-by coordinates will be encountered. Therefore, it can perform processing that involves object interdependence (e.g. summing object coverage for points) for points in a sliding window not wider than the maximum object size. To put it short, ,,localized reduction" phases in algorithms are possible: two objects can mutually influence a computational result only if they overlap.

Furthermore, such a limited model of computation lends itself for parallelization. In our execution model tasks declare two procedures: one for independent processing of objects and one for a ,,localized reduction" (finishing) step. The objects are presented to a task in chunks, a set of as many objects as requested by the task itself. Object processing is in theory independent for every chunk and can thus be scheduled for separate threads in parallel; the finishing step is serial.

Figure 2 B illustrates this on an example. The upper chart shows which step is performed by which thread for any given chunk, while the lower one is a classical sequence diagram of the same execution scenario. Note that while the ,,process" step is always executed on a pre-defined number of objects, the ,,finish" step is executed once regardless of how much the ,,sweeping pointer" has advanced since the last finish step. This may be scheduled to any thread and may take arbitrarily long (as opposed to ,,process" steps with a constant input chunk size). In this example, Thread #1 finishes the region between the first object of Chunk 1 and the last one of Chunk 2, while Thread #2 finishes the region between the first object of Chunk 3 and the last object of Chunk N.

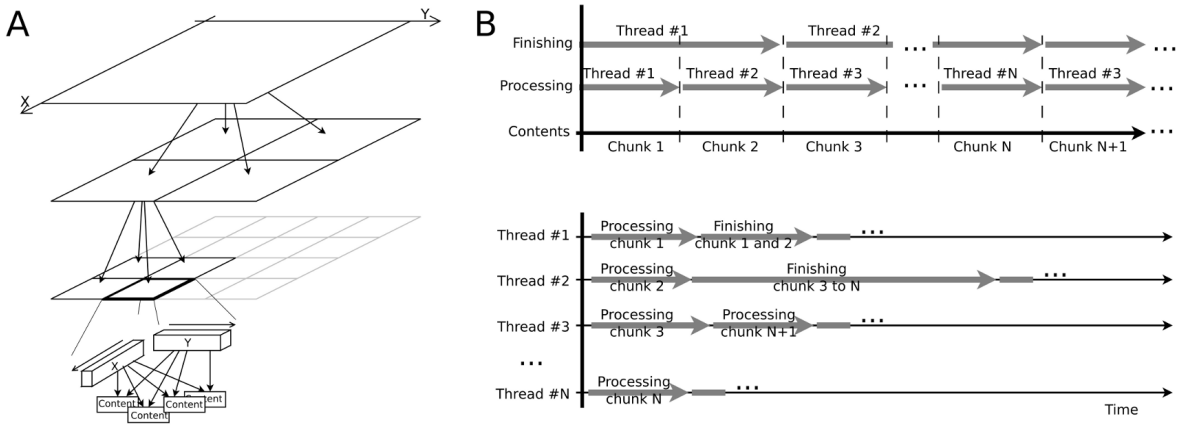


Figure 2: Schematic view of the data cache model and the task execution model

C. *Performance Analysis*

The task we used for performance testing was an actual algorithm of the genomics project. We created a discretized coverage map of the chart regions with a given resolution, essentially summing the number of overlapping evidences for each discrete block. The evidences of our data set were trapezoids with longer parallel sides of 2000 base pairs. The region of the genome under investigation covered a square with sides of about 5.5 megabasepairs, where 90,199 evidences were located. The square sizes we tried were 100, 200, 400, 500 and 1,000 basepairs; the smaller size means finer resolution.

The algorithm first generates the discrete blocks for each object (which is fully parallelizable), then it stores these into global memory (for which it competes with other threads). In the finishing step (which is sequential) it sums the generated discrete blocks and stores them back into the tree.

We first executed the algorithm single-threaded for all resolutions and measured the ratio of time spent with parallel, competing and serial tasks (Figure 3 A). The theoretical maximum speedup is limited by Amdahl's law [3] (1); however, it does not take into account the possible blocking of competing operations. We therefore evaluated this formula for two cases: assuming no blocking and assuming fully serial execution. The theoretical maximum is therefore between these two values.

$$S_{max} = \frac{1}{1 - P + \frac{P}{N}} \tag{1}$$

We performed the measurements for 2, 3 and 4 threads on a quad-core Intel i7 processor (Figure 3 B, C and D). It is clearly visible that the system schedules the task rather efficiently in the middle values of the parameter range, with speedups nearing the best-case maximum. The inferior performance in the extremities of the parameter range is yet to be explained; our hypothesis is that in the 100 basepair case the higher overhead of object creation and garbage collection degrades performance, while on the other end, the total execution time may be too short to take advantage of parallel execution.

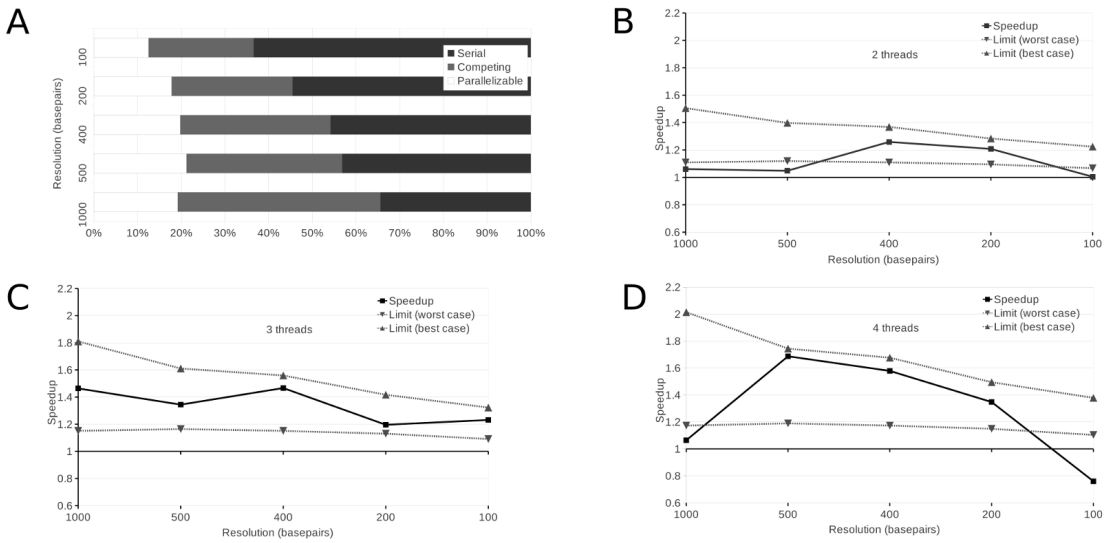


Figure 3: Measurement results

References

[1] M. Stratton, P. Campbell, and P. Futreal, "The cancer genome," *Nature*, 458(7239):719–724, 2009.

[2] S. Sindi, E. Helman, A. Bashir, and B. Raphael, "A geometric approach for classification and comparison of structural variants," *Bioinformatics*, 25(12):i222, 2009.

[3] G. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, spring joint computer conference*, pp. 483–485. ACM, 1967.

NETWORK MODELING OF LEARNING IN SCHIZOPHRENIA*

Mihály BÁNYAI

Advisors: Péter ÉRDI, Fülöp BAZSÓ, György STRAUSZ

I. Introduction

Schizophrenia is a complex disorder that manifests on many different levels ranging from the physiology of single neurons through the dynamics of neural circuits to cognitive, affective or behavioral symptoms (for clinical characterization of the disease see [1], for a review on functional connectivity during schizophrenia see [2]). We use a multi-level modelling approach to connect these different levels. Our previous results demonstrated the schizophrenic patients show a significant impairment in object-location associative learning tasks [3]. Here we developed a neural network model incorporating models of brain regions involved in paired-associate learning in order to analyze the mechanisms underlying behavioral differences between schizophrenic patients and control subjects.

II. Methods

A. The experiment

We used a paired-associate learning paradigm in which subjects are required to learning arbitrary associations between locations (in space) and objects (with unique identities). In the encoding phase, subjects see which object is asssociated to which field. In the recall phase, subjects are presented with cues in the fields, and they have to give a verbal answer specifying the corresponding object. The raw fMRI data reflect the blood-oxygene-level changes in the brain in 4 dimensions, with a time resolution in the order of seconds and spatial resolution in the order of 10 mm^3. The data is normalized, and in order to capture the large-scale dynamics of the cortex, regions of interest (ROI) are selected based on the anatomical location of brain areas related to associative learning (primary visual cortex (V1), superior parietal (SP) and inferior temporal cortex (IT), hippocampus (HPC) and dorsal prefrontal cortex (PFC)). This way we obtain five time series describing the activities of the selected areas during the experiment, in addition of the time series of the experimental conditions and the behavioral data containing the subjects' answers to the memory cues.

B. Neural network model

A feed-forward network creates the representations of the identity of the object and its location in the model of the areas IT and SP in the ventral and dorsal visual streams, respectively. The proposed role of the hippocampus is to bind these two representations together so that when cued by the location, the correct object can be recalled. The model is outlined in Figure 1, where the detailed hippocampal circuitry is also given, including the dantate gyrus (DG) and the CA3 region. Moreover, in order to model cognitive control, we included a prefrontal region which controls learning and recall processes presumably by modulating the plasticity and the efficiency of hippocampal synapses.

The interaction of the areas is described by the following equations:

*This summary is based on M Bányai, B Ujfalussy, V Diwadkar and P Érdi. Impairments in the prefronto-hippocampal interactions explain associative learning deficit in schizophrenia. BMC Neuroscience 12(Suppl 1):93, 2011.

$$a^i_{dg} = \sum_j w^{ji}_{sp\to dg} r^j_{sp} + \sum_k w^{ki}_{it\to dg} r^k_{it} \tag{1}$$

$$r^i_{dg} = F(a_{dg}, \gamma_{dg}) \tag{2}$$

$$\gamma_{dg} = \frac{\sum_i (r^i_{dg})^2}{N_{DG}(\sum_i r^i_{dg})^2} \tag{3}$$

$$\tag{4}$$

where a^i_{dg} denotes the activation of unit i in the dentate gyrus, γ stands for sparseness, r for the firing rate, w for the synaptic weights, $F()$ is a treshold linear function and N_{DG} stands for the number of cells in the dentate gyrus. Similarly, the activation of the CA3 region and the inferior temporal coretx is given by the following equations.

$$a^i_{ca} = R\sum_j w^{ji}_{dg\to ca} r^j_{dg} + L\sum_j w^{ji}_{sp\to ca} r^j_{sp} \tag{5}$$

$$a^i_{it} = L\sum_j w^{ji}_{ca\to it} r^j_{ca} \tag{6}$$

$$\tag{7}$$

Here L and R are random variables sampled between 0 and 1 according to the learning and recall phases of the task. Switching between learning and recall phase distributions of R and L models the effect of the prefrontal cortex on the circuitry, which implements the decision if we have to update the stored memory patterns or we have to retrieve one of them.

The synaptic weights are updated according to a Hebbian learning rule dependent on the phase we are currently in. All weights are updated in a similar fashion to the following equation:

$$\Delta w^{ij}_{sp\to dg} = (1-R)\alpha r^j_{dg}(r^i_{sp} - w^{ij}_{sp\to dg}) \tag{8}$$

C. Dynamic causal modelling (DCM)

DCM provides a complete phenomenological model framework for the analysis of fMRI data. For a detailed description see [4]. The model structure consists of two components: a neural state equation and a hemodynamic model. The neural component describes the time evolution of the neural state variables, x, which refer to the neural activity of the brain areas. The input variables, u, are the conditions defined by the experiment (Eq. 9). The connectivity parameters of the neural model are the elements of the three matrices, $\theta_n = \{A, B, C\}$. A contains the intrinsic coupling parameters, the causal effects of the areas on each other, B contains the modulatory parameters, the effects of the inputs on the intrinsic connections, and C contains the direct effects of the inputs on the areas.

$$\dot{x} = (A + \sum_{i=1}^{N} u_j B^j)x + Cu \tag{9}$$

$$y = \lambda(x, \theta_\lambda) \tag{10}$$

The hemodynamic component, λ, describes the nonlinear mapping from the neural activity to the fMRI signal, y, actually measured in the brain areas (Eq. 10). For the details see [5]. We need to

estimate the values of the parameter set, $\theta = \{\theta_h, \theta_\lambda\}$ best fitting to measurement data. One possible procedure to do so is the Bayesian maximum a posteriori (MAP) estimation technique defined by Eq. 11, where M denotes the specific connectivity pattern of the model.

$$p(\theta \mid y, M) = \frac{p(y \mid \theta, M)p(\theta \mid M)}{p(y \mid M)} \tag{11}$$

For all probability distributions in 11, we assume that both the prior ($p(\theta \mid M)$) and posterior ($p(\theta \mid y, M)$) distributions are Gaussians, and the MAP estimation is defined as the mean of the posterior distribution. To compare models with different connectivity patterns, we can set the prior probability of having certain connections is a certain model to zero.

The model evidence is the probability of obtaining the actual measurement conditioned on the model form integrated over parameter space. This way we obtain the expected posterior probability of each model regarding the subject group. For a complete description of the comparison method see [6].

$$p(y \mid M) = \int p(y \mid \theta, M)p(\theta \mid M)\, d\theta \tag{12}$$

III. Results

A. Simulation of the neural model

We fitted the model performance to the behavioral data. On Figure 2 the blue circles indicate the healthy subjects' performance, the red circes the patients'. The empty circles show the simulated data for two different parameter settings for the distributions of L and R. Our model predicts that the impairment of cognitive control of the prefrontal cortex over hippocampal processes implies inaccurate regulation of hippocampal dynamics and explains the poorer performance of patients in this task.

For a more complete discussion of the results see [7].

B. Estimation of the dynamic model

The comparison of different model connectivity structures lead to the finding that in schizophrenia, the task-related functional network is fundamentally different relative to healthy controls, the patients' networks lacking connections in the control signal flow from the prefrontal area to the lower level areas (Figure 3). Comparison of the parameter estimates in the two groups implies significant impairments in the prefrontal control of hippocampal memory formation in patients (Figure 4). This finding supports the results from the neural network simulations by identifying the neural basis of the learning-recall decision impairments.

For a more complete discussion of the results see [8].

IV. Conclusion

Our results show that prefronto-hippocampal interactions are material in understanding learning impairments in schizophrenia, and our multi-level approach is suitable to integrate the explanatory capabilities of mechanistic neural models with the analytical power of data-driven phenomenological approaches.

Acknowledgement

We are thankful to the National Institutes of Mental Health, the Children's Research Center of Michigan, the Elizabeth Elser Doolittle Investigator-ship and the Henry Luce foundation.

Figures

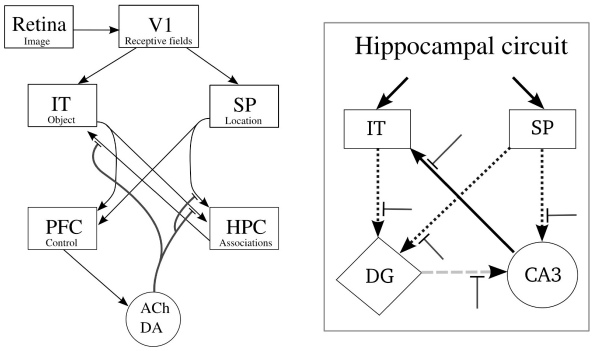


Figure 1: The outline of the neural network model. The rigth panel gives the details of the prefronto-hippocampal circuit and its interaction with the superior parietal and inferior temporal areas.

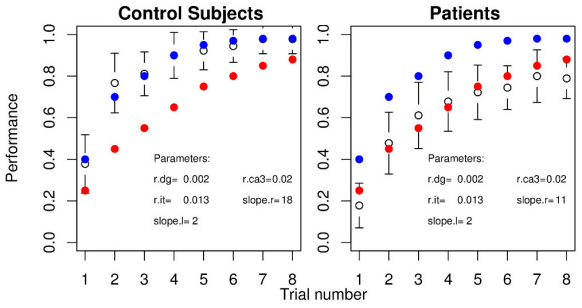


Figure 2: Results of the simulation of behavioral data by the neural model. Performance means the average ratio of correct answers given by the subjects in each trial.

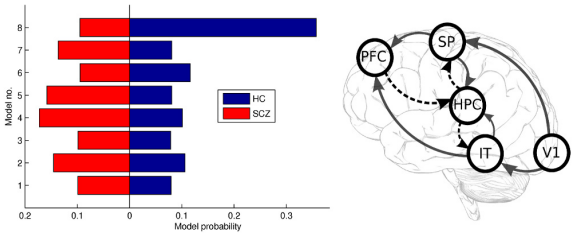


Figure 3: Results of the comparison of the DCM models. Prefronto-hippocampal, hippocampo-inferior temporal and hippocampo-superior parietal interactions are less likely to be present in the patient group.

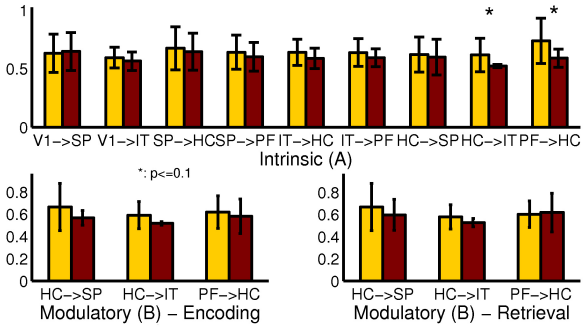


Figure 4: Results of the parameter level comparison between healthy (yellow) and schizophrenia (red) groups. Largest differences are seen in the strength of the interaction between the prefrontal cortex and hippocampus and the hippocampus and the inferior temporal cortex.

References

[1] P. J. Harrison, "The neuropathology of schizophrenia. a critical review of the data and their interpretation," *Brain*, 122:593–624, 1999.

[2] M. E. Lynall, "Functional connectivity and brain networks in schizophrenia," *Journal of Neuroscience*, 14:9477–87, 2010.

[3] V. Diwadkar, B. Flaugher, T. Jones, L. Zalányi, B. Ujfalussy, M. S. Keshavan, and P. Érdi, "Impaired associative learning in schizophrenia: Behavioral and computational studies," *Cognitive Neurodynamics*, 2:207–219, 2008.

[4] K. J. Friston, L. Harrison, and W. D. Penny, "Dynamic causal modelling," *NeuroImage*, 19:1273–1302, 2003.

[5] K. J. Friston, A. Mechelli, R. Turner, and C. J. Price, "Nonlinear responses in fmri: The balloon model, volterra kernels and other hemodynamics," *NeuroImage*, 12:466–477, 2000.

[6] K. E. Stephan, W. D. Penny, J. Daunizeau, R. Moran, and K. J. Friston, "Bayesian model selection for group studies," *NeuroImage*, 46:1004–1017, 2009.

[7] P. Érdi, V. Diwadkar, and B. Ujfalussy, "The schizophrenic brain: A broken hermeneutic circle," *Neural Network World*, 19:413–427, 2009.

[8] M. Bányai, V. Diwadkar, and P. Érdi, "Model-based dynamical analysis of functional disconnection in schizophrenia," *NeuroImage*, 58:870–877, 2011.

CALCULATION OF SINGLE NEURON'S CURRENT SOURCES (SUMMARY OF PHD WORK IN 2011)

Dorottya CSERPÁN
Advisors: Gábor HORVÁTH, Zoltán SOMOGYVÁRI

I. Introduction

In this paper we present the application of an electrophysiological method based on two different neuron models. The sCSD (spike Current Source Density) method [1] uses extracellular potentials recorded in mouse brain to evaluate the current flow through the cell membrane on different parts of the cell. The traditional CSD analysis [2] calculates current source densities denoted to layers of the neocortex, the novelty of sCSD method is the calculation of the current source densities of single cells from the measured extracellular potentials. The relationship between these two quantities is given by Poisson-equation. In matrix formalism, Φ vector contains the measured n extracellular potentials generated by the n current sources (C vector) and the so called transfer matrix (T) modells the relationship between extracellular potentials and current sources. The current source densities can be calculated by multiplying the extracellular potentials with the inverse (or pseudoinverse) of the transfer matrix.

$$\mathbf{C} = \mathbf{T^{-1}\Phi} \tag{1}$$

To make the solution unique the cell-electrode distance has to be estimated. Here two application of sCSD for two different cases are shown, one regarding to neurons, which can be approximated with a line segment and an other for the spherically symmetric cells.

II. The spike Current Source Density analysis of different neurons

A. Linear segment approximation

The line segment approximation can be used for elongated neurons which are parallel with the electrode. By positioning the electrode perpendicular to the brain surface, the pyramidal cells (1.a) of the neocortex will satisfy this assumption. These cells are represented as n point sources arranged in a line segment, where n is equal to the number of electrodes. In this case the elements of the transfer matrix (T) are the following:

$$T_{ij} = \frac{1}{4\pi\sigma d_{ij}}, \tag{2}$$

where d_{ij} is the distance between the i^{th} electrode and j^{th} point source and σ is the electrical conductivity of the extracellular medium. The cell-electrode distance was determined by the introduction of a measure, which has an extremal value at the real distance. On the sCSD distributions (1.c) white colour indicates the flow of positive ions into or negative ions outwards the cell. The strong white blob at 2 ms is the action potential initialization by the soma and the bright blobs afterwards in the neighbouring segments are probably the dendritic backpropagation.

B. Spherical shell approximation

Some of the cells have a spherically symmetric morphology (e.g. relay cells (1.d)): the soma is in the centre and the dendrites form a ball around it. It's worthwhile to calculate the current sources of sphere shells, since these corresponds to inputs from different brain regions. The distance of the electrode and the soma was set to 20 μm, hence there is no method to predict it yet. We can specify this arrangement as following: the electrode is 50 μm far from the soma, the others are in a line perpendicular to the

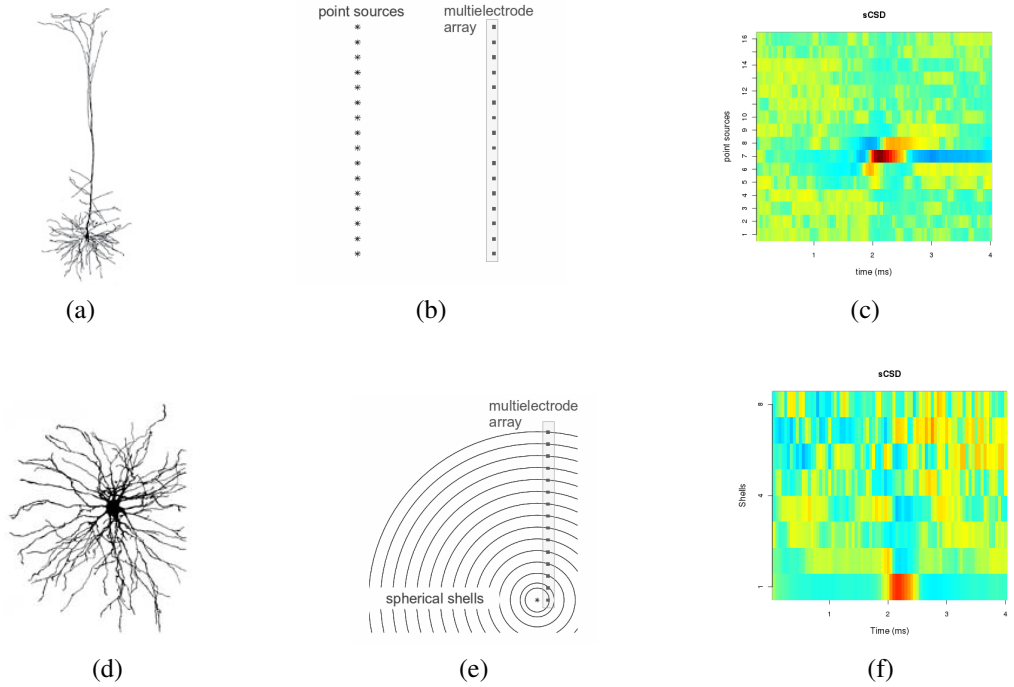


Figure 1: (a) pyramidal cell (b) linear segment model (c) sCSD distribution of a pyramidal cell (d) relay cell (e) spherical shell model (f) sCSD distribution of a relay cell

line between the soma and the first electrode. Every electrode corresponds to one spherical shell. The electrode can measure the extracellular potentials of current sources of the inner shells. In this case the transfer matrix has the following form:

$$T_{ij} = \frac{1}{4\pi\sigma d_i} \text{ if } j \leqslant i \tag{3}$$

$$T_{ij} = 0 \text{ if } j > i \tag{4}$$

where d_i is the distance between the i^{th} electrode and the soma. The action potential initialization is also recognizable here (1.f), the darker blobs before it on the outer shells might are caused by the input currents.

III. Conclusions

By the application of sCSD method various interesting phenomena can be observed, which cannot be seen in other in vivo extracellular measurements. Still there are limitations in the usage, further investigations and the development of the models are needed. The future goals are the investigation of the origins coming to a neuron and effect of the inputs on firing.

Acknowledgement

I am grateful to László Acsády and Péter Barthó for providing the data.

References

[1] Z. Somogyvari, L. Zalanyi, I. Ulbert, and P. Erdi, "Model-based source localization of extracellular action potentials," *J. Neurosci. Meth.*, 147(2):126–137, 2005.

[2] C. Nicholson and J. A. Freeman, "Theory of current source density analysis and determination of conductivity tensor for anuran crebellum," *J. Neurophysiol.*, 38(2):356–368, 1975.

IMPUTATION AND HAPLOTYPE RECONSTRUCTION IN GENETIC ASSOCIATION STUDIES (SUMMARY OF PHD WORK IN 2011)

Peter SARKOZY
Advisor: Peter ANTAL

I. Genetic Association studies

The quickly decreasing cost of performing genotype analysis on more and more samples and loci have ushered in an era where even smaller research groups can perform genetic association studies. These studies are quickly becoming limited by univariate association analysis [1], as most single-gene diseases and mutations have already been tied to known mutations. Univariate analysis does not provide an efficient and knowledge rich method of uncovering multiple interactions and pathway level overviews.

Partial genetic association studies (PGAS) are commonly performed with the aim of uncovering associations with a particular phenotype or measurable trait by determining the genotypes of a set of loci selected on genes that are suspected to be a part of the biological pathway which plays a role in determining the phenotype. Genome wide association studies (GWAS) genotype millions of tag SNP's per genome, where the tag SNPs are selected to provide maximum linkage disequilibrium based coverage of the genome. While GWAS studies have a lower price per SNP genotyped, they require thousands of samples while measuring millions of loci. PGAS studies present a more focused, narrow search of associations with a specific trait.

II. The genetics of trait impulsivity

The genotyping capabilities provided by the Semmelweis University enabled us to design a genetic association study to map the underlying genetics of trait impulsivity. Trait impulsivity is a complex construct which is measured through a questionnaire called the Barratt Impulsivity Scale [2] (BIS), containing 30 questions which are subdivided into three groups, which measure three components of impulsivity, non-planning, motor impulsivity and cognitive impulsivity.

We used the genotyping system to measure 96 single nucleotide polymorphisms that covered 16 genes along various neurotransmitter pathways. The SNP's were selected in order to provide maximum coverage, while also including SNP's that were previously shown to be associated with trait impulsivity.

A Probabilistic genotyping

The measurements provided a low call rate which is typical of most genotyping systems, in our case there were eight SNP's which did not return any recoverable data. The remainder of the samples had a 79% call rate. The high number of missing samples made it impossible for us to run Bayesian multilevel analysis [3] on the data set, because the method requires a complete set of variables for each sample. Discarding samples with one or more SNP's missing would have resulted in an extremely small data set. I investigated the underlying cause of the high failure rate for each measured SNP, and found that some of the results were recoverable through manual clustering. I also found that data concordance was around 99% even in the case of the high quality measurements.

I was able to utilize the fact that measurements were not of tag SNP's only, but were rather intended as a high coverage mapping of the measured genes. Using the linkage disequilibrium (LD) present between loci that were in physical proximity (Fig. 1.), it is possible to recover most of the failed measurements.

I developed a tool that allows the analysis of the raw measurement data, which is recorded as a digital image. First the high-frequency noise is eliminated from the image while the background noise level is calculated as well, then the best well alignment is found using an image difference based classifier. Low-frequency noise components, such as wipe marks, specks of dust and residual chemicals are then removed by checking for the evenness of intensity inside each well. After subtracting both low and high frequency noise components I obtained accurate intensity data for the entire set, as well as quality parameters which gave a distribution of possible genotypes for each sample. This intensity data was in concordance with the intensity data supplied by the measurement system itself, only showing marked differences where there were low-frequency noise components and physical artifacts visible on the measurement plates.

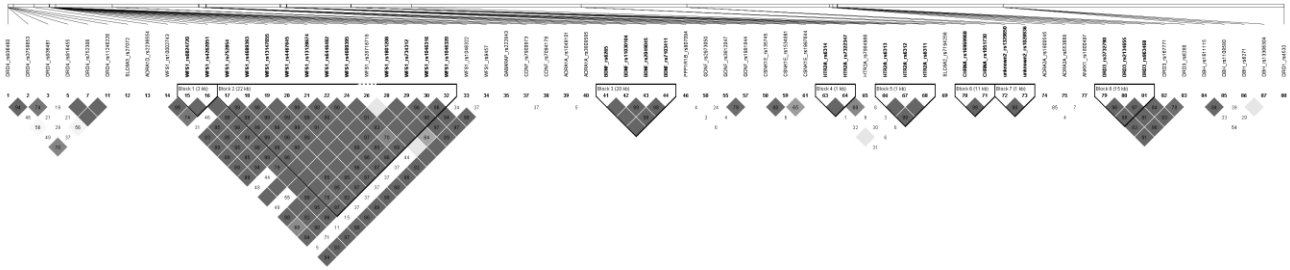


Figure 1. LD for an entire set of measurements, 96 SNPs across 16 genes. The image represents a covariance matrix with its main diagonal plotted horizontally. Dark squares mark high linkage.

B Recovery using LD

Using the software package Impute2 [4], which incorporates multiple heterogeneous data sources, including data from the 1000 Genomes project [5] as well as the HapMap project, it is possible to impute missing genotypes if we have measured other SNP's which are in linkage disequilibrium.
I tested the software by imputing a single SNP without supplying any measurement information about the SNP to the impute software package. I then combined the output of impute with the accurate intensity data obtained by my image processing algorithm, and the two independent results showed a remarkable concordance (Fig 2.). This meant that even though the clusters on most of the failed measurements were hard to separate, they still contained valid and useful information.

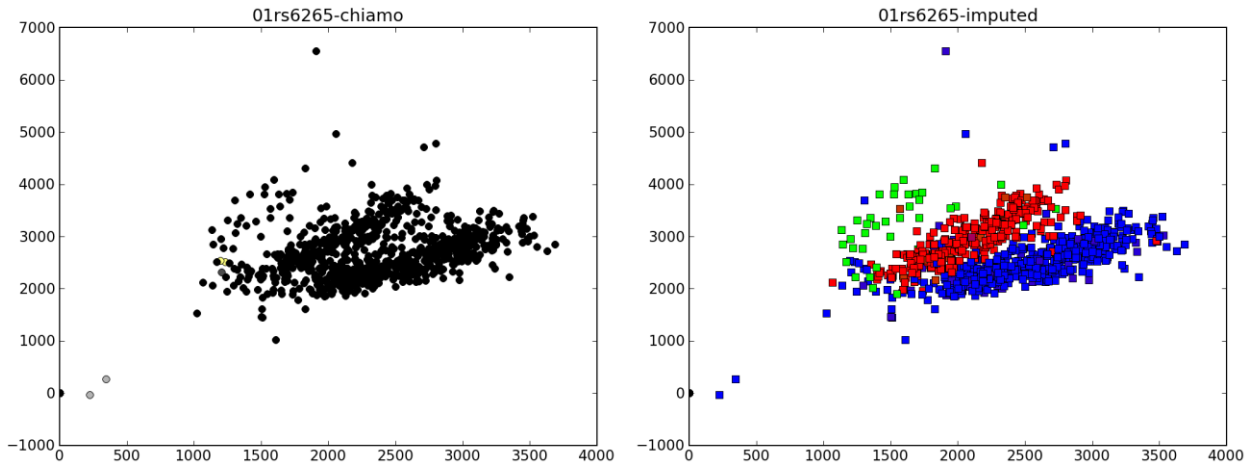


Figure 2. The left image plots the green and blue intensity data from a single SNP. The right plot shows the fusion of imputed values (RGB) plotted independently of the intensity data.

III. Analysis

Analysis of this recovered data set presented multiple problems when I attempted to use Bayesian networks in Bayesian multilevel analysis to uncover the strongly relevant associations between the

SNP's and the impulsivity target variables. The Bayesian method searches in the space of graph structures, and for computational reasons each node has a limited number of parents. The imputed data contained many SNP's in tight linkage, which meant that if one SNP was selected as a parent of a target variable while searching in the space of graph structures, all the other linked SNP's would also enter as parents. This violated the limit on the maximum number of parents for each node, resulting in inconsistent results.

A Tag SNP selection

Multiple sets of tag SNP's were selected using various r^2 values. This method was used to overcome the instability of the MCMC algorithm [6] by trimming redundant and tightly linked variables from the data set. Analysis of these results were most stable with an r^2 value of 0.5. Unfortunately this resulted in reducing our initial set of 96 SNP's to 52. Much of the information loss in this way flattened the posterior probabilities of association for the analysis (Fig. 3.).

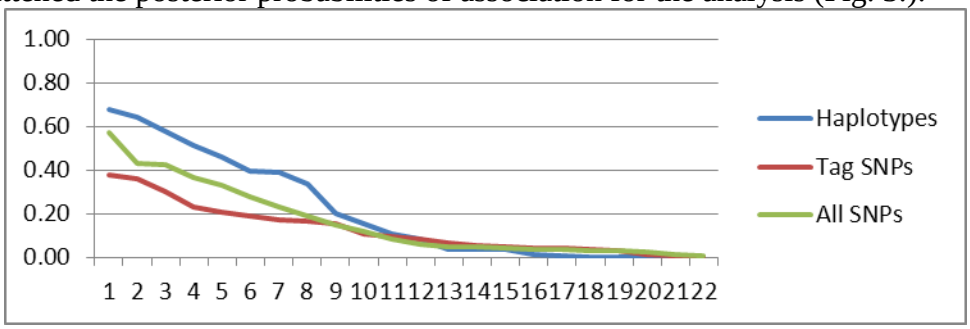


Figure 3. The Y axis shows the MBM posterior probabilities, while the X axis contains the genes sorted in descending order. Tag SNP's show the flattest posteriors, while the haplotype level shows a more peaked distribution.

B Haplotype reconstruction

The central concept in converting SNP's to haplotypes is centered around exploiting the linkage disequilibrium that exists between close loci to transform sets of SNP's to haplotypes. The sets are most often defined by the genes that the SNP's are located on. In case of large genomic regions spanning more than 50 kb, that are likely to be in separate haplotype blocks (the haplotype blocks can be identified with software like HaploView [7] each block is defined as a separate set), allowing separation of non-tightly linked regions.

Haplotype reconstruction provides a reduction in the dimensionality of the data [8], because SNPs inside a haplotype block are represented as a single variable, at the cost of an increase in cardinality. SNPs have a cardinality of 3 (homozygous wild, heterozygous, homozygous mutant), where the theoretical cardinality of a haplotype block is 4^n, where n is the number of SNP's in the block. The limited number and approximate stability of recombination hotspots creates linkage disequilibrium. This results in mutations being passed on together. Exploiting this linkage allows us to greatly reduce the resulting cardinality.

The question whether a heterozygous SNP has a mutant allele on the paternal or the maternal chromosome (the phase of the SNP) can also be resolved with haplotype reconstruction – but only on the haplotype level. This allows us to get a complete picture of the mutations on both copies of the gene in the genome. Inter-gene phase remains unknown, but it does not have a major effect.

Haplotype reconstruction also allows us to aggregate rare variants in a knowledge rich fashion [9], as well as allowing higher-level aggregations such the gene or pathway levels.

C Data averaging in Bayesian model averaging

Haplotypes are at a higher abstraction level than singular SNP's and the ability to incorporate phase information into a genetic association study can prove very useful. Bayesian model averaging

is used to overcome the limitations presented by moving to a higher abstraction level, as well as allowing us to use probabilistic phased data.

We use a probabilistic genotyping model $p(D_N)$ based on image processing and clustering. Next we apply an existing phasing method to generate probabilistic phased genotype data $p(D'_N|D_N)$. Finally, as unifying framework we overview the use of uncertain datasets in statistical data analysis, which is summarized as Eq. (1).

$$p(\alpha(M)) = E_{p(D_N)}[E_{p(D'_N|D_N)}[E_{p(M|D'_N)}[\alpha(M)]]] \qquad (1)$$

This shows the embedded averaging over genotyping uncertainty and phasing uncertainty, and additionally Bayesian model averaging (assuming that $\alpha(M)$ denotes an important feature of model *M*, e.g. a direct causal relation between two variables). Averaging Markov blanket membership values is trivial, while averaging over phasing uncertainty presents a unique challenge in averaging the model features such as Markov blanket sets and Markov boundary graphs.

IV. Results

Fusion of uncertain measurements with the linkage disequilibrium known from publicly available data makes it possible to recover most of the failed measurements, while also allowing us to quantify the uncertainty present in our recovery.

Figure 3 shows that the posterior probabilities are the most peaked when using haplotype reconstruction while also showing that the loss of information when using only tag SNP's is unacceptable. Data averaging results in a large increase in the computational requirements for the BMV MLA method, as it requires the sampling of the sources of uncertainty and then running multiple instances of the analysis. Depending on the number of samples produced, this ranges from 10-100 fold increase in computational time, limiting this option to cases where there is a high level of aggregation, a low number of variables or if sufficient computational resources are present.

The results of my research are under review for publication in the journal Artificial Intelligence in Medicine. My work is supported by TAMOP-4.2.1/B-09/1/KMR-2010-0002, TAMOP - 4.2.2.B-10/1--2010-0009, and the following Hungarian Scientific Research Funds: OTKA-PD-76348 and NKTH TECH 08-A1/2-2008-0120 (Genagrid).

References

[1] J. McClellan and MC. King. Genetic heterogeneity in human disease. Cell, 141:210-217,2010.

[2] J. H. Patton, M. S. Stanford, E. S. Barratt, Factor structure of the barratt impulsiveness scale, J Clin Psychol 51:768-774, 1995

[3] P. Antal, A. Millinghoffer, G. Hullam, Cs. Szalai, and A. Falus. A Bayesian view of challenges in feature selection: Feature aggregation, multiple targets, redundancy and interaction. JMLR Proceeding, 4:74-89, 2008.

[4] B. Servin, M. Stephens, Imputation-based analysis of association studies: candidate genes and quantitative traits, PLoS Genetics 3(7):e114, 2007.

[5] The 1000 Genomes Project Consortium: A map of human genome variation from population-scale sequencing Nature 467:1061–1073, 2010.

[6] S. Chib and E. Greenberg. Understanding the metropolis-hastings algorithm. The American Statistician, 49(4):327-335, 1995.

[7] J. C. Barrett, B. Fry, J. Maller, M. J. Daly, Haploview: analysis and visualization of LD and haplotype maps, Bioinformatics 21:263-65, 2005.

[8] Clark A.G. Inference of haplotypes from PCR-amplified samples of diploid populations. Molecular Biology and Evolution, 7(2):111-122, 1990.

[9] M. Stephens and P. Donnelly. A comparison of Bayesian methods for haplotype reconstruction from population genotype data. The American Society of Human Genetics, 73(5):1162-1169, 2003.

A Model-driven Framework for Guided Design Space Exploration*

Ábel HEGEDÜS
Advisor: Dániel VARRÓ

I. Introduction

Design space exploration (DSE) is a the analysis of several "functionally equivalent" implementation alternatives, which meet all design constraints in order to identify the most suitable design choice (solution) based on quality metrics such as cost or dependability. Design space exploration is a challenging problem in application areas (such as dependable embedded systems and IT system management), where model-driven engineering (MDE) techniques are already popular. DSE can be performed either during the design process to find optimal designs or during runtime to help dynamic reconfigurations.

In traditional DSE problems, the design constraints and quality metrics are numeric attributes to express cost, time or memory limits, etc. However, systems with modular software and hardware architectures (like AUTOSAR [2] in the automotive domain or large reconfigurable architectures) led to the introduction of complex restrictions on the graph-based model of the system.

Existing DSE approaches usually apply model checking complemented with exhaustive state space exploration or solve finite domain constraint satisfaction problems that cannot effectively handle structural constraints and dynamic manipulation of elements. In order to alleviate these issues, designers often provide additional information (hints) about the system (e.g. from earlier experience or by some analysis) that can reduce the design space to a more feasible size.

Guided model-driven design space exploration aims to explore alternative system designs efficiently by making use of advanced model-driven techniques (e.g. incremental model transformations) and hints (obtained by analysis tools or provided by the designer). These hints are interpreted during the exploration to continue along promising search paths (using selection criteria) and to avoid the traversal of unpromising designs (by cut-off criteria). Additionally, the use of incremental techniques leads to exploration strategies that are able to find additional (alternative) solutions.

In this paper, a model-driven framework for guided design space exploration is described, where the system states are graphs, operations are defined as graph transformation rules, while goals and constraints are defined as graph patterns.

II. Overview of the Approach

The schematic overview of the framework for guided design space exploration is illustrated in Figure 1. First, the design problem description specifies the domain where the exploration takes place to produce solutions. It includes: (1) the initial state of the system at the start of the exploration, (2) the set of manipulation operations (called labeling or exploration rules) defined on the system, (3) goals described as structural or numerical constraints, which need to be satisfied by solution states found by the exploration, and (4) global constraints, which are satisfied by the initial and solution states and all intermediate states on the trajectory between them.

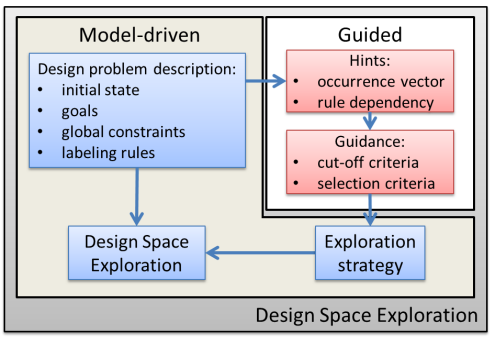


Figure 1: Model-driven Guided DSE

*This summary is based on the ASE'11 paper with the same title [1]

The design space exploration performs the search for solutions by exploring the design (or state) space of the problem description. It starts from the initial state and traverses reachable states by applying the operations on the system. In order to find a solution quickly, exploration is often aided by an exploration strategy. A simple strategy (as proposed in [3]) may use random selection in a depth first search or statically assign priority levels to operations. However, a more advanced strategy should also determine whether a given state will never lead to a valid solution (i.e. it is a dead end) and states reachable from it should not be traversed. In a guided approach, the exploration strategy relies on guidance, which uses hints for driving the traversal and identifying dead ends.

Hints are information originating from the designer or (as in [4]) from some automated analysis carried out using formal methods that often abstract the design problem description. For example, the result of such analysis can be information regarding the number of operation applications (called occurrence vector). The guided approach uses occurrence vectors and dependency relations between rules, computed from pre- and postconditions, as hints (see [1]).

Finally, the guidance calculates and interprets hints and provides decision support for the exploration strategy (see details in [1]). In this approach, guidance is defined as the evaluation of cut-off and selection criteria based on the current state and the hints (as defined in [4]). Cut-off criteria identify dead end states and bound the exploration, while selection criteria prioritize available rules in a state by their likelihood of leading to a final (solution) state.

III. Exploration Strategy

Guided exploration strategies can be categorized by the used hints and guidance. Here, two guided strategies are presented, the first uses occurrence vectors only as hints (occurrence), while the other uses rule dependency as well (full guidance). Note that the full guidance strategy uses rule priorities only if two labeling rules were evaluated as equal by the guidance. These strategies are compared to the fixed priority strategy.

Figure 2 illustrates the design space exploration for these techniques on a simple example. The circles denote the traversed states which are numbered according to the traversal order, while the applicable rules are listed beside them.

Downward arrows illustrate rule applications, while dotted arrows represent backtracking from invalid or cut-off states. The same rule can be applied multiple times at a given state if more than one applicable match is found in the graph (see state 2 on the right side). The exploration terminates when an optimal solution is found. A solution is optimal if the path leading to it contains the least number of rule applications (i.e. it is the shortest trajectory to a solution model).

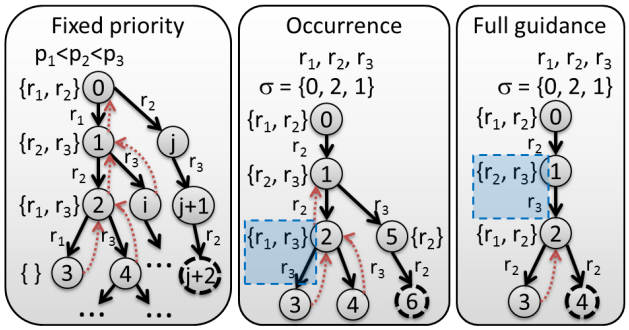


Figure 2: Comparison of exploration strategies

In the case of the fixed priority strategy, the next applied operation is the one with the highest priority among the applicable ones. As the depth-first technique is used in the fixed priority exploration strategy, the first solution found by that strategy is often several times longer than the optimal, suboptimal solutions are used iteratively as depth limits to force the exploration to find shorter solutions.

The *occurrence* strategy applies operations based on the occurrence vector provided by the system analysis. Figure 2 shows that the hint states the r_2 should be applied twice and r_3 once. Therefore, r_1 is not applied in state 0 or 2 (highlighted) in order to be compliant to the occurrence vector.

The full guidance exploration strategy (illustrated in the right side of Figure 2) takes the dependency

relations between rules into account in addition to the occurrence vector. Therefore, in state 1 (highlighted) it selects r_3 for the next application. Rule r_2 is applicable on two matches in state 2, the first leading to a dead-end state, while the second application leads to a solution in state 4. Note that the selection in state 1 leads to a reduced traversed design space compared to the occurrence strategy.

IV. Implementation Details

Figure 3 gives an overview of the implemented guided design space exploration framework. The implementation uses the VIATRA2 model transformation framework [5], which provides metamodeling capabilities and supports model transformations based on the concepts of graph transformations and abstract state machines. Its incremental pattern matcher is used as a powerful query engine.

The design space exploration is performed by the constraint satisfaction engine, CSP(M), presented in [3], where rules, goals and constraints (specified using graph transformation rules and patterns) are used in solving constraint satisfaction problems over the input model (both included in the design problem description).

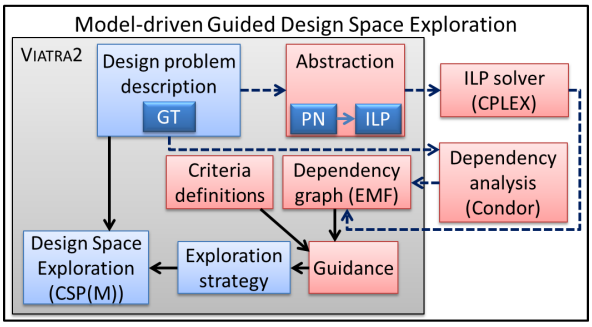


Figure 3: Overview of the guided DSE framework

The abstraction of graph transformation rules into Petri nets (PN) and Integer Linear Programming (ILP) problems are also automated. The industry leading IBM CPLEX [6] optimization tool is used, which supports the calculation of alternate solutions (occurrence vectors used for initializing the dependency graph G_d). The edges of G_d are computed from the transformation rules using the Condor [7] dependency analyzer tool, while the graph itself is built and stored as an Eclipse Modeling Framework (EMF) instance model. The criteria definitions and the criteria evaluation algorithm (guidance) are implemented in Java as separate components, and are connected to the guided design space exploration strategy.

V. Evaluation of the Approach

The aim of the evaluation was to demonstrate that the full guidance strategy is more efficient than the other strategies (namely, fixed priority and occurrence strategy, which were used for previous measurements in [3]) as it traverses considerably fewer states and does not introduce significant overhead, thus provides better runtime in the other approaches for most of the experiments.

Cases used in the Evaluation

For evaluation, the cloud case study presented in [1] and a service configuration case study (presented in [8]) were used. These cases are relevant in the context of model-driven DSE as they represent both design and runtime exploration problems and make comparison with previous results [8, 3] possible.

Both case studies included multiple cases (see Figure 4). PowerOn cases deal with empty initial models, while Reconfigure (RC) cases deal with existing models which must be modified to satisfy goals. Finally, the Clustered DB case requires databases to be deployed on clusters.

Evaluation Environment and Method

The evaluation was carried out 5 times for each test case and strategy in the following way:

(1) the initial model is loaded into VIATRA2, (2) the goals, constraints and operations are added to the framework, (3) the exploration component is initialized and runtime measurement is started (using wall time with OS-level nanotime precision). Next, (4) the design space exploration framework computes solutions. Finally, (5) the runtime measurement is stopped and the results are saved. The exploration is limited to 1 million visited states.

Evaluation of Results

The graph in Figure 4 shows the results of measurements using the case study models. For each case, the length of the shortest solution trajectory (the number of applied rules) is given in parenthesis with the average number of visited states during the design space exploration illustrated for each strategy.

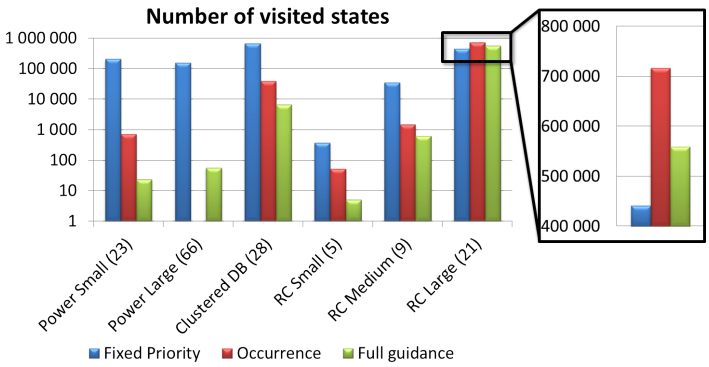


Figure 4: Results for exploration

The following observations were made based on the results from the experiments: (1) The combined use of occurrence vectors and rule dependency for cut-off and selection criteria based guidance outperforms our previously published strategies; (2) The added computation required for criteria evaluation does not significantly increase runtime; and (3) The under-approximation of the occurrence vector based analysis ensures that guided strategies always find optimal solutions first (similarly to the A* algorithm). Note that in the last experiment the fixed priority strategy visits less states than the guided strategies because of a high number of infeasible occurrence vectors.

VI. Conclusion and Future Work

Guided DSE exploration uses hints to reduce the number of states traversed when searching for solutions. Hints are used (i) to identify dead-end states (cut-off criteria) and (ii) to order applicable rules in a given state (selection criteria). In this paper, I summarized the results of developing a model-driven framework for guided DSE, which uses rule dependency and occurrence vectors as hints for the exploration strategy. Evaluation of the exploration strategies using a cloud configuration problem showed that the criteria-driven approach reduces the design space further thus increasing efficiency. The framework was also successfully applied for generating quick fixes for domain specific modeling languages [9], evaluation with BPMN business process models showed that it can work as a good design time development assistance tool.

Future work aims to improve the framework by introducing incremental techniques for identifying states, handling guidance and space exploration. There are also plans for extending the framework to handle EMF models.

References

[1] Á. Hegedüs, Á. Horváth, I. Ráth, and D. Varró, "A model-driven framework for guided design space exploration," in *26th IEEE/ACM Intl. Conf. on Automated Software Engineering (ASE 2011)*. IEEE Computer Society, 11/2011 2011.

[2] AUTOSAR Consortium, *The AUTOSAR Standard.*, `http://www.autosar.org/`.

[3] Á. Horváth and D. Varró, "Dynamic constraint satisfaction problems over models," *Software and Systems Modeling*, 2011, 10.1007/s10270-010-0185-5.

[4] Á. Hegedüs, Á. Horváth, and D. Varró, "Towards guided trajectory exploration of graph transformation systems," *ECEASST*, 40, 2010, Petri Nets and Graph Transformation 2010.

[5] A. Balogh and D. Varró, "Advanced model transformation language constructs in the VIATRA2 framework," in *ACM Symp. on Applied Computing (SAC 2006)*, p. 1280–1287, Dijon, France, 2006. ACM Press.

[6] IBM, *ILOG CPLEX Optimizer*, `http://www.ibm.com/software/integration/optimization/cplex-optimizer/`.

[7] ROOTS, *Condor*, `http://roots.iai.uni-bonn.de/research/condor/`.

[8] S. Varró-Gyapay and D. Varró, "Optimization in Graph Transformation Systems Using Petri Net Based Techniques," *ECEASST*, 2, 2006, Petri Nets and Graph Transformation 2006.

[9] Á. Hegedüs, Á. Horváth, I. Ráth, M. C. Branco, and D. Varró, "Quick fix generation for dsmls," in *IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC 2011*. IEEE Computer Society, 09/2011 2011.

DYNAMIC BACKWARD SLICING OF MODEL TRANSFORMATIONS*

Zoltán UJHELYI
Advisor: Dániel VARRÓ

I. Introduction

Model-driven design (MDD) aims to simultaneously improve quality and productivity by providing early model validation and automating various phases of software development including source code, test case or configuration generation. Model transformations (MT) play a central role in automating such tasks. Model-to-model transformations take one (or more) source model(s) as input and derive one (or more) target model(s) as output typically together with detailed traceability links.

Model transformations are captured in the form of a MT program, which can be taken as a regular piece of software. However, elementary steps in MT programs are captured by highly data-driven declarative rules, while complex transformations are assembled from elementary steps using some imperative control structures. Due to this hybrid nature of MT languages, the direct adaptation of existing software engineering results is problematic, especially, when designing complex MTs where debugging and validation plays a crucial role.

Many integrated development environments (IDEs) include sophisticated program slicing techniques to calculate control and data dependencies between the statements of a program. When debugging MTs, transformation experts would require similar support to identify parts of the MT program which have causal dependence on a selected statement of the MT program (called slicing criterion). However, the slicing criterion of a MT program can also depend on an element of the underlying model when a read or write operation on the element causes a causal dependency. For instance, declarative model queries issued by transformation rules might introduce data dependencies that can only be detected precisely by creating relevant slices of the transformed models as well.

In a previous short paper [1], we introduced the concepts of model transformation slicing for the first time (up to our best knowledge), and identified the main scientific challenges. We argued that the adaptation of existing program slicing techniques turns out to be non-trivial as MT programs take models as an additional input. Therefore, slices of a MT program should simultaneously incorporate the causally dependent statements and the causally dependent parts of the underlying model.

A further difference to a traditional (dynamic) program slicing setup comes from the fact that MTs are executed mostly at design time in modern IDEs, which frequently save additional information when executing a transformation in order to provide undo/redo support. Consequently, execution traces of a MT run are readily available for transformation MT slicing.

In [1], we informally sketched the idea of dynamic backward slicing of model transformations. In the current paper, we provide an overview of our approach together with an extensive experimental evaluation of our dynamic backward slicing approach that is carried out using various case studies taken from previous model transformation benchmarks.

II. Model Transformation Slicing: An Example

A traditional program slice consists of the parts of a program that (potentially) affect the values computed at some point of interest [3]. In [1] we defined the slicing problem of MTs as illustrated in Figure 1. The slicing algorithm receives three inputs: the slicing criterion, the model transformation program, and the models on which the MT program operates. As an output, slicing algorithms need to

*This summary is based on the following publications [1, 2].

produce (1) transformation slices, which are statements of the MT program depending or being dependent on the slicing criterion, and (2) model slices, which are parts of the model(s) depending (or being dependent) causally on the slicing criterion (due to read or write model access).

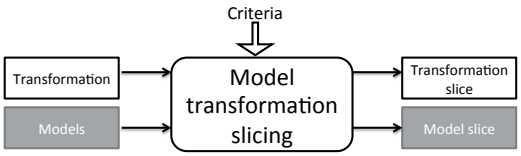


Figure 1: Slicing Problem of Model Transformations

There are various program slicing approaches in the literature, in this paper we focus on dynamic and backward slicing. Dynamic slicing relies on a specific execution (test case) of the program. In case of MT slicing, the affected statements of the slicing criterion are calculated wrt. this specific execution, while model slices can be identified on the specific (input) models. A backward slice of a MT consists of (1) all statements and control predicates of the program and (2) all elements of the underlying model the slicing criterion is dependent on.

In the remaining part of this section, we informally present the core technique of dynamic backward slicing of MT programs using a demonstrative example of Petri net simulation formalized by model transformations in the MT language of the VIATRA2 framework. This example is frequently used to demonstrate how model transformations can be used in model simulation scenarios, furthermore, it already served as a performance benchmark for MTs [4].

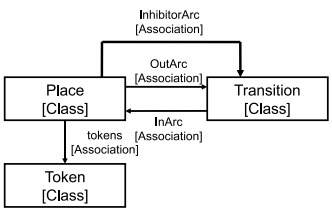
A. *Running Example: Simulation of Petri nets*

Petri nets are bipartite graphs with two disjoint set of nodes: Places and Transitions. Places can contain an arbitrary number of Tokens that represent the state of the net (marking). The process called firing changes this state: a token is removed from every input place of a transition, and then a token is added to every output place of the firing transition. If there are no tokens at an input place of a transition, the Transition is disabled, and thus it cannot fire. The structure of the modeling language of Petri nets is formalized by a corresponding metamodel in Figure 2a.

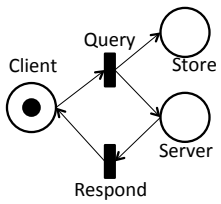
Example 1. Figure 2b depicts a simple Petri net. The net consists of three places, representing a Client, a Server and a Store and two transitions. If the Client issues a query (the Query transition fires), the query is saved in the store (a token is created), the Server gets the control (another token is created), and the client waits for a response (the Respond transition fires and a token is removed).

B. *The* VIATRA2 *Transformation Language*

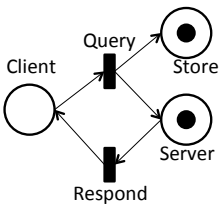
Graph patterns are often considered as atomic units of MTs [5]. They represent complex structural conditions (or constraints) that are to be fulfilled by a part of the (input) models. Graph patterns are also used to declaratively define model manipulation steps.



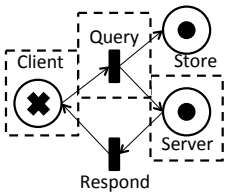
(a) Metamodel for Petri nets



(b) A Simple Petri net



(c) The Model after Firing the Query Transition



(d) The Calculated Model Slices

Figure 2: A Petri net Example

Example 2. The sourcePlace pattern (Line 1) in Listing 3 is used to identify the source places of a transition. The pattern consists of a Transition node Tr and a Place node Pl connected by an edge of type of OA. It is important to note that as only variables Tr and Pl appear in the header of the pattern, they can be received from the caller of the pattern as input parameters, or passed back to the caller as output parameters, while the variable `OA` is an internal pattern variable.

Graph transformation (GT) provides a high-level rule and pattern-based manipulation language for graph models. GT rules are specified using a left-hand side graph (pattern) to decide the applicability of the rule, and a right-hand side graph (pattern) which declaratively specifies the result model after the rule application. This is achieved by removing all elements only present in the LHS, creating all elements only present in the RHS, and leaving every other element unchanged.

Example 3. Listing 3 presents two simple GT rules that are (respectively) used to add a token to or remove a token from a place. The LHS pattern of the addToken (Line 19) pattern consists of a single place, while its RHS extends it with a token and an edge. This means, applying the rule creates a token and connects it to the place.

Complex MT programs can be assembled from elementary graph patterns and graph transformation rules using some kind of a control language. In our examples, we use abstract state machines for this purpose as available in the VIATRA2 framework with all the necessary control structures including the sequencing operator (seq), ASM rule invocation (call), variable declarations and updates (let and update constructs), if-then-else structures, and single or simultaneous application at possible matches (forall and choose).

Example 4. The fireTransition rule (Line 27) in Listing 3 describes the firing of a transition in VIATRA2. At first the code determines whether the input parameter is a fireable `Transition` using the `isTransitionFireable` pattern. Then in a sequence the GT rule removeToken is called for each sourcePlace, followed by a call to GT rule addToken for every targetPlace.

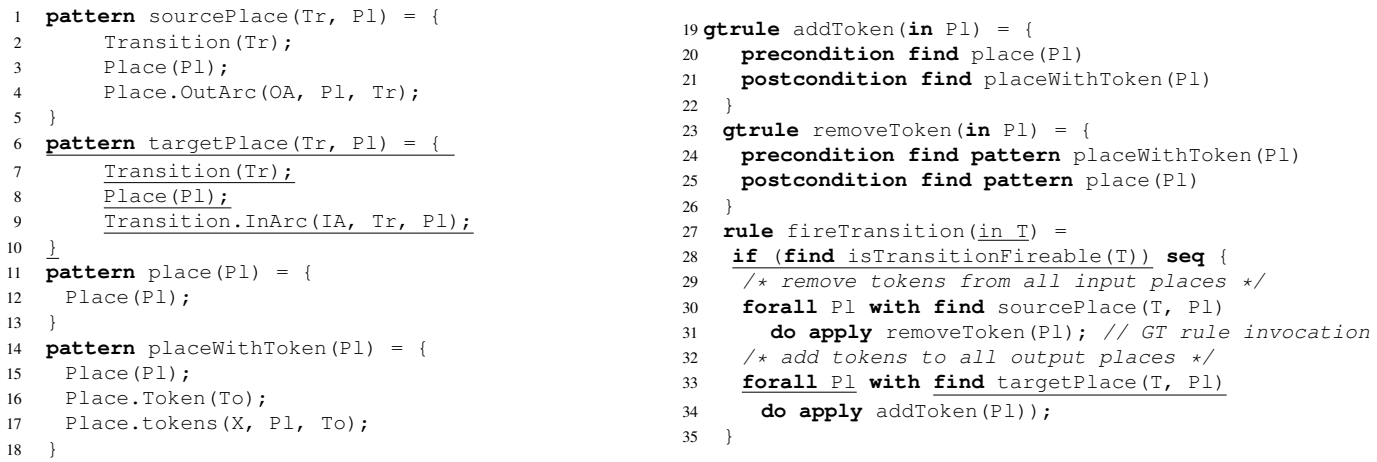
C. A Sample Dynamic Backward Slice

To illustrate the MT slicing problem, we consider the execution of the rule fireTransition called with the transition Query as a parameter. Figure 2c displays the model after the execution: the token from the place Client is removed, while a token is added to places Store and Server. As a slicing criterion, we selected the invocation of the GT rule addToken in Line 34 together with variable Pl.

We can calculate the backward slices for the criterion as follows. (1) At the last item of the trace the variable Pl is bound during the matching of the pattern targetPlace, so the pattern invocation is part of the slice (Line 33). (2) As the pattern matching of targetPlace uses model elements (Server, Query and the IA between them), they have to be added to the model slice. (3) The forall rule in Line 33 is included in the slice as it defines the variable Pl. (4) On the other hand, the token removal operation (Line 31) does not affect the slicing criterion as Pl is a different variable (redefined locally), T is passed as input parameter, while no model elements touched by this GT rule are dependent from those required at the slicing criterion. (5) Although the condition in Line 28 does not define variables that are used later in the slice, it is added because the execution of the forall rule added to the slice depends on the evaluated condition. (6) Finally, as the slice includes statements that use the variable T, its definition as an incoming parameter of the fireTransition rule is added to slice.

The calculated program slice is represented by underlined source statements in Listing 3: the pattern targetPlace and parts of the fireTransition rules are included.

As model elements are created and deleted during the execution of the transformation, the corresponding model slice can contain elements from multiple states. To display the slice, Figure 2d shows the final model state of the Petri net by adding the deleted token from the place Client as a cross. In this figure the model slices are included inside the dashed rectangles: the transition Query, the places Client and Server, the arcs between the included transition and place, and the token in Server.



```
1  pattern sourcePlace(Tr, Pl) = {
2       Transition(Tr);
3       Place(Pl);
4       Place.OutArc(OA, Pl, Tr);
5  }
6  pattern targetPlace(Tr, Pl) = {
7       Transition(Tr);
8       Place(Pl);
9       Transition.InArc(IA, Tr, Pl);
10 }
11 pattern place(Pl) = {
12   Place(Pl);
13 }
14 pattern placeWithToken(Pl) = {
15   Place(Pl);
16   Place.Token(To);
17   Place.tokens(X, Pl, To);
18 }
19 gtrule addToken(in Pl) = {
20   precondition find place(Pl)
21   postcondition find placeWithToken(Pl)
22 }
23 gtrule removeToken(in Pl) = {
24   precondition find pattern placeWithToken(Pl)
25   postcondition find pattern place(Pl)
26 }
27 rule fireTransition(in T) =
28  if (find isTransitionFireable(T)) seq {
29   /* remove tokens from all input places */
30   forall Pl with find sourcePlace(T, Pl)
31     do apply removeToken(Pl); // GT rule invocation
32   /* add tokens to all output places */
33   forall Pl with find targetPlace(T, Pl)
34     do apply addToken(Pl));
35 }
```

Figure 3: Slicing criterion: variable Pl in Line 34

III. Evaluation and Future Work

The aim of the evaluation is to demonstrate that our MT slicing approach provides small, relevant slices describing both control and data dependencies from the selected slicing criteria wrt. to the corresponding execution trace. To illustrate the simultaneous slicing of the models and MT programs, we selected four fundamentally different MT programs (two model simulation and two model-to-model transformations) in [2] available in the VIATRA2 transformation framework, which were already used in the past for performance benchmark investigations (e.g. at various model transformation tool contests). These MT programs are used "as is", without any manual changes to them.

Our measurements showed that our MT slicing approach is able to provide small and understandable slices that encapsulates both control and data (model) dependency information simultaneously. In addition to that, by the analysis of the measurements we concluded that (1) in case of complex, imperative control structures, MT slices are dominantly created from the dependency between the statements of the transformation programs, thus traditional imperative program slicing techniques are applicable and provide appropriate results. However, (2) in case of declarative transformation rules slices are primarily created based on model dependencies.

In the future, we plan to investigate other slicing challenges for MT programs. The overview of the MT slicing problems in [1] provides a full research agenda to address both dynamic and static slicing, in forward and backward directions. Future research may also investigate how relevant breakpoints can be automatically inserted during MT debugging where slicing techniques can also be exploited.

References

[1] Z. Ujhelyi, A. Horváth, and D. Varró, "Towards dynamic backwards slicing of model transformations," in ASE 2011, 26th IEEE/ACM International Conference on Automated Software Engineering. ACM, 2011.

[2] Z. Ujhelyi, A. Horváth, and D. Varró, "Dynamic backwards slicing of model transformations," in ICST 2012, Fifth International Conference on Software Testing, Verification and Validation, 2012, Accepted.

[3] F. Tip, "A survey of program slicing techniques," Journal of Programming Languages 3(3), pp. 121–189, 1995.

[4] G. Bergmann, Á. Horváth, I. Ráth, and D. Varró, "A benchmark evaluation of incremental pattern matching in graph transformation," in Proc. 4th International Conference on Graph Transformations, ICGT 2008, pp. 396–410, 2008.

[5] D. Varró and A. Balogh, "The model transformation language of the VIATRA2 framework," Sci. Comput. Program., 68(3):214–234, 2007.

MAPPING TOPIC MAPS TO COMMON LOGIC

Tamás DEMIÁN
Advisor: András PATARICZA

I. Introduction

This work is a case study for the mapping of a particular formal language (Topic Map[1] (TM)) to Common Logic[2] (CL). CL was intended to be a uniform platform ensuring a seamless syntactic and semantic integration of knowledge represented in different formal languages. CL is based on first-order logic (FOL) with a precise model-theoretic semantic. The exact target language is Common Logic Interchange Format (CLIF), the most common dialect of CL. Both CL and TM are ISO standards and their metamodels are included in the Object Definition Metamodel[3] (ODM). ODM was intended to serve as foundation of Model Driven Architecture (MDA) offering formal basis for representation, management, interoperability, and semantics. The paper aims at the evaluation of the use of CL as a fusion platform on the example of TM.

II. The Topic Maps

TM is a technology for modelling knowledge and connecting this structured knowledge to relevant information sources. A central operation in TMs is merging, aiming at the elimination of redundant TM constructs. TopicMapConstruct is the top-level abstract class in the TM metamodel (Fig. 1). The later detailed ReifiableConstruct, TypeAble and ScopeAble classes are also abstract. The remaining classes are pairwise disjoint.

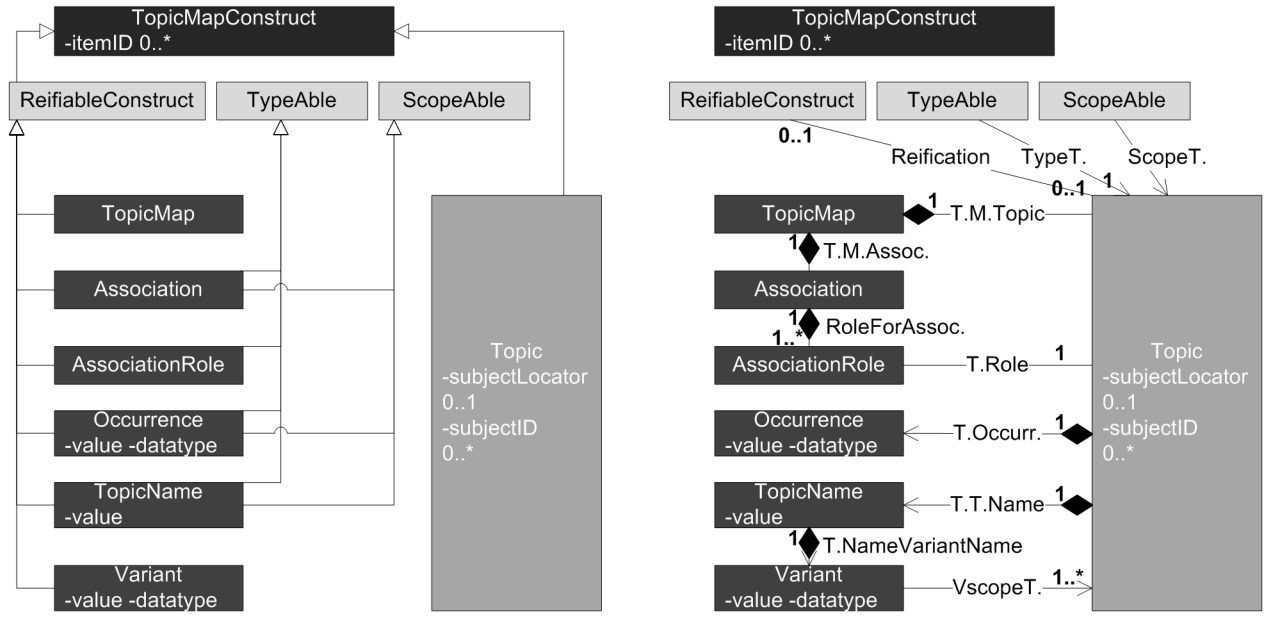


Figure 1: The class hierarchy and the relation and attribute names of the TM metamodel.

TopicMap is a set of topics and associations. Topic is a symbol used within a TM to represent exactly one subject, in order to allow statements to be made about that subject. The Association and AssociationRole classes enable TM to express hyperrelations between topics. The number of contained AssociationRole instances is the arity of the relation.

The Occurrence class represents a relationship between a subject and an information resource. Occurrences are essentially specialized associations, where one participant in the association can be an information resource. For example the topic 'Iron' may have an occurrence with Unicode string attributes: ($datatype = IRI$, $value = http://en.wikipedia.org/wiki/Iron$) or an other occurrence with ($datatype = density$, $value = 7.88g/cm^3$).
In many cases the extensibility of TM constructs by additional information is useful, for example by adding occurrences to an association, or by assigning a name to an occurrence. Reification is the act of making a topic represent the subject of another TM construct in the same TM. For example, creating a topic that represents the relationship expressed by an association is reification. Reification may store meta-data like authors, version number, copyright external documentation or schema. Typeable constructs must have a type topic. Statements can be declared with occurrences. Topic names, variant names, occurrences, and associations are statements, whereas assignments of identifying locators to topics are not considered as statements. All statements have a scope, representing the context within which a statement is valid. Scope could be for example the source of information or solution of a homonymic conflict (e.g., 'file' in the context of fishes or IT).
Subject identifiers and subject locators of topics enable the use of references to internal or external information resources. The identifiers may represent the subject of the topic in a human readable form without any specific semantic. On the other hand, the locator is unique and conforms some locator notation standard like URI or IRI. Subjects which are not information resources should be treated as subject identifiers.
A topic name consists of the base name, and variants of it, known as variant names. Topic names may have a scope, which defines in what context the topic name is an appropriate label for the subject. Suitable base names for people, countries, and organizations are their names, while base names for documents, musical works, and movies might be their titles. Essentially, a base name is a specialized kind of occurrence. A variant name is an alternative form of a topic name that may be more suitable in a certain context than the corresponding base name, so variants must have a more specific scope.

III. The mapping

The ODM contains both the mapping from TM to Web Ontology Language (OWL) and the mapping from OWL to CL. So the composition of mappings seemed to be a long but steady way to perform the desired mapping. We followed this way to be as close to the spirit of ODM as possible. However, we faced many problems during this procedure. The understanding of the transformations using different formalisms (QVT, translation tables and corresponding axioms) was the most time consuming part of the job. The cited mappings of ODM[3] are incomplete. It is not surprising, because the syntax and semantic of OWL and CL are relatively complex.

The OWL-CL mapping of ODM is based on the work of P. Hayes[4] which is still incomplete. Many OWL and RDF statements lack a clear CL translation. Fortunately, the first mapping does not result in ambiguous statements or constructs, so the composition of the mappings was relatively simple. Hayes mentions two logical styles of mappings between formal languages: translation and embedding. Let us see the next OWL/RDF triple:

```
behind rdf:type owl:TransitiveProperty
```

which carries the meaning that 'behind' is a transitive binary relation. This is semantically equivalent with the next CLIF sentence:

```
(forall (x y z)(implies
    (and (behind x y)(behind y z))
    (behind x z)
))
```

However, the next statement is also a correct CLIF sentence with the same meaning but the OWL/RDF syntax survives somehow:

```
(rdf_triple behind rdf:type owl:TransitiveProperty)
```

We will refer to the first solution as translation while the second -which preserves the vocabulary of the source language- is called embedding. Embedding can be considered as a specific syntactic sugar. Therefore we need some extra axioms in CLIF that fix the meaning of them:

```
(forall (x y z)(iff (rdf_triple y x z)(x y z)))
(forall (x y)(iff (rdf:type x y)(y x)))
(forall (p)(iff
    (owl:TransitiveProperty p)
    (forall (x y z)(implies (and (p x y)(p y z))(p x z) ))
))
```

If some semantic aspects of the source language were unclear, we could express it using embeddings without sugar axioms.

```
( forall (u) (   implies (subclass u)(superclass u)   ))
```

where the unary relations above are type-instance relations. This notation in [4] [5] is just a natural convention but not a semantic extension.

```
(forall (p q)(iff
    (supsub p q)
    (forall (x)(   implies (q x)(p x )   ))
))
```

Returning to the subject, the composition of TM-OWL and OWL-CL mappings was followed by a consolidation of the result (Table 1). The introduced CL relations should inherit their semantics from TM.

Table 1: Mapping between TM and CL constructs

TM construct		TM construct parameters	CL constructs
Topic Map	x:TM		CL Module
Association I.	x:A	parent y:TM	x:Sentence of y:Module
Reification		x:TM reified by y:T(parent z:TM)	Importation(z) into x, (= x y)
ID-s and locators			CL Names
multiple ID-s		...	(= id1 id2)
sup.t.-subt. relation		x:T, y:T, ...	(supsub x y)
type-inst. relation		x:T, y:T, ...	(x y)
Association II.	x:A	type is y:T	(y x)
AssociationRole	x:AR	type y:T, parent z:A, player q:T	(y z q)
Occurence	x:O	type y:T, parent z:T, value q:string	(y z q)
TopicName	x:TN	type y:T, parent z:T, value q:string	(y z q)
Variant	x:V	parent z:T, value q:string	(variant z q)
Scope		x:TMC, scope is y:T	(scope x y)

We will demonstrate the mapping by the following TM example (Fig. 2) followed by the corresponding CL theory. Example includes different styles of type-instance and subtype-supertype relations because ODM uses both of them. The TM and CL Module containment is neglected here.

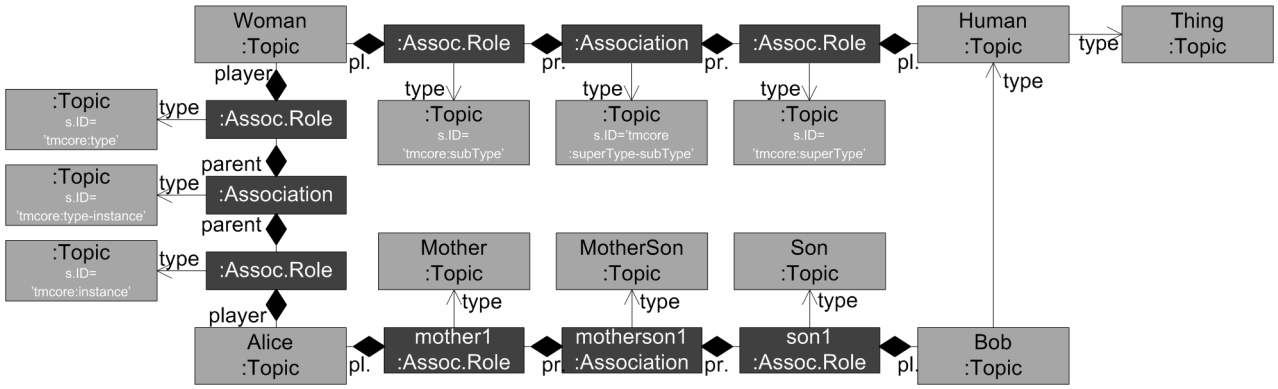


Figure 2: TopicMap example (pr.=parent, pl.=player).

```
(supsub Human Woman)
(supsub Thing Human)
(Woman Alice)
(Human Bob)
(MotherSon motherson1)
(Mother motherson1 Alice)
(Son motherson1 Bob)
```

The mother-son relation is quite verbose but includes the role names of the arguments too. We can easily shorten it as follows:

```
(MotherSon Alice Bob).
```

IV. Conclusions

Despite the fact CL aims to be a common platform for semantic integration and covers the semantic of FOL, it does not cover the most common semantic relations used in knowledge representation like type-instance, subtype-supertype or containment. Type-instance relations have a natural notation as an unary relation and the supertype-subtype relation can be derived from it using the features of FOL. However, these relations should be handled explicitly as semantic extensions[2] in a new dialect or should be standardized. There are no available CL examples for fundamental features, e.g., for importing modules, exclusion sets, texts. The lack of complete and normative translations from the most important knowledge representation languages is the main obstacle of CL to become a widely accepted common platform. Correct usage of CL texts, modules and importation remained an open question. It is crucial in matching ontologies but there are no relevant examples at all, e.g., in [3][4][5]. Neuhaus [6] shows that the semantics of modules are erroneous and suggests two options how to fix them. So CL alone is insufficient to express the basic concepts of knowledge representation in a uniform way and suffers from semantic inconsistencies.

References

[1] ISO, *ISO/IEC JTC1/SC34 Topic Map Reference Model*, 2008, available at http://www.isotopicmaps.org/sam/sam-model/.

[2] ISO, *ISO/IEC 24707:2007(E) Common Logic (CL): a framework for a family of logic-based languages*, 2007, available at standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip.

[3] OMG, *Ontology Definition Metamodel*, 2009, available at www.omg.org/spec/ODM/1.0/.

[4] P. Hayes, *Translating Semantic Web languages into Common Logic*, 2005, draft, available at http://www.ihmc.us/users/phayes/cl/sw2scl.html.

[5] P. H. C. Menzel, "A new axiomatic semantics for semantic web languages," .

[6] F. Neuhaus, "The semantics of modules in common logic," in *Proc. of the Third Interdisciplinary Ontology Meeting*, vol. 3, Tokyo, Feb. 27–28 2010.

CONTEXT-BASED REQUIREMENTS REPRESENTATION FOR SOFTWARE TESTING (SUMMARY OF PHD WORK IN 2011)

János OLÁH
Advisor: István MAJZIK

I. Introduction

The quality and the success of software systems depends on how well it meets its intended needs of the stakeholders. Requirements engineering (RE) is a process which involves the understanding the needs of stakeholders; understanding the context in which the system will operate; capturing the requirements into an adequate format; and validating that the documented requirements match the negotiated requirements.

Beyond these core activities, many software engineering activities and artefacts are based on the requirements, e.g., software testing. In this paper we introduce a novel requirements description method, which is able to capture requirements related to the context of software (i.e., requirements, that can be expressed solely in terms of the context), and can effectively support the generation of functional test cases.

II. Requirements engineering

By definition, requirement is "a condition or capability needed by a user to solve a problem or achieve an objective" (ISO/IEC/IEEE 24765:2010; Systems and Software Engineering). While other software engineering activities result in artefacts directly affecting the software's behaviour (i.e., they are part of the solution), requirements engineering rather defines the problem that has to be solved. This means that functional requirement descriptions are written entirely in terms of the environment of the desired software.

In RE related research papers, RE research is decomposed into five tasks: elicitation, modelling, requirements analysis, verification and validation, and requirement management [1]. These RE activities are usually iterative, since they involve many actors (stakeholders, architects, developers, testers, etc.) with different background and focus.

These activities cannot be executed and understood separately. In this paper we want to emphasize how an adequate requirements description can help generating sound functional test cases, thus we will focus on requirements elicitation and modelling.

Requirements elicitation involves activities to understand the goals and motives of building the desired software system. Activities of elicitation comprise discussion with the stakeholders, hence informal and intuitive models are especially popular, like use-cases, scenarios (simple text, cartoons or video), demos and simulations. These promise easy comprehension and quick feedback by non-professional stakeholders.

During requirements modelling phase, the high-level models constructed for stakeholders are refactored to more detailed, precise, often formal models, that models can be used by software architects and developers to plan and create the software system. Scenario-based models are usually constructed during this phase of RE. These models contain a "step-by-step description of series of events, that may occur concurrently or sequentially" (IEEE 1362-1998 (R2007)). Scenario-based models are well-known examples of relationship between RE and software testing. For example UML sequence diagrams are popular to model the flow of messages, events and actions between the components of a system (in addition, they are still relatively easy to understand by non-technical stakeholders). Scenario-based

models contains all necessary information (i.e., input data and conditions, and expected output data and conditions), thus testers can reuse these scenario descriptions to derive test cases.

III. Context modelling

In this paper we propose a novel requirements representation approach, which utilizes the model of the software's context. Contextual RE techniques analyse stakeholders' requirements with respect to a particular context. The context is the environment in which the desired software will operate. The importance of context during requirements elicitation has long been recognized by RE research community, however in many cases the contextual techniques often focus on improving the elicitation phase (e.g., stakeholder interview design) by understanding the context [2].

In contrast to this approach, we refer to context modelling as the main part of the elicitation phase. In most cases, there are assumptions about the context, and in many cases this context is finite and entirely known. During the modelling of the context we create a metamodel that contains every possible object that may appear in the environment of the software, and every action through which the environment can affect the software.

We would like to express the functional requirements in terms of this metamodel. This means that the requirements are model instances conforming to this metamodel. Again, this representation is only appropriate in case of context-related functional requirements, where the user needs can be described entirely with objects and actions from the software's environment.

For example in case of an autonomous robot vacuum cleaner, the environment is a household. The stakeholders have the apriori knowledge of every possible object that may appear in the environment of the robot (e.g., furniture, living beings), and the possible actions are known as well (e.g., perceptions/actuations of the robot). Another example is a software with web-based interface, with given GUI objects (buttons, text boxes, combo-boxes, drop-down lists, etc.) and standard actions (e.g., click, double-click, draw, type, etc.) that can be executed on these objects.

In order to thoroughly express the software's functionalities with model instances, we need to capture the expected output as well. This means we need to complement the metamodel or create another metamodel with actions and objects, which define the output of the software. It is important to note that in many cases the input and output metamodels may be the same or at least the objects may be similar, and only the actions are different. For example in the household environment of the autonomous vacuum cleaner the objects are common for both metamodels, but the actions may be different for the output metamodel (e.g., start draining, give alert). In the other example, the output actions will contain elements associated with the server side action (e.g., page loaded, exception thrown).

Using this approach to RE elicitation, we express the software functionalities as ordered pairs of input and output models.

A. Motivation of context-based requirements representation

According to the context representation described above, during the elicitation phase we create a structured view of the software's environment that contains every relevant object, action and relationship among these, instead of an ad-hoc representation of context elements in different requirements without capturing their hierarchy and relations. In addition we may assign arbitrary attributes to these model elements (e.g., timing to actions, position to objects).

Beyond the requirements, using the domain-specific, structured metamodel of the context enables the representation of the necessary constraints, that for example require the presence of a certain object, or define cardinality restrictions.

The primary aim of the proposed representation is to facilitate test case derivation. We represent functional test data as model instances of the context metamodel that contain the requirements input model instances, and test executions are evaluated by searching for the output model instances. We

think that the proposed representation is advantageous in the following cases:

- To support automated test case derivation with instantiation of elements of the metamodel (i.e., model instance generation) and fulfilment of the constraints.
- To derive robustness test cases through the violation of constraints.
- To systematically combine requirements (i.e., input models of different requirements) to derive more complex test cases.
- To detect incompleteness and explore inconsistency among the requirements on the basis of the metamodel.
- To define precise coverage metrics based on the hierarchy of elements and their relations.

B. Example

Let us consider the example of a web-based software. On the login screen there are two text boxes (e.g., login name and password) and one button (e.g., login). Thus the metamodel in this simple case contains two text boxes and a button, and a two actions, namely type and click. Additionally there are actions executed by the software, for example "page load". This simple metamodel describes the context of the program (i.e., in this case we composed one single metamodel).

A basic requirement to express may be the following: when the user provides the login credentials and then click on login button, the welcome screen is loaded.

In order to describe this requirement, we instantiate two text boxes and two type actions from the input metamodel, and associate each text box instance with two type action instance. The instances can have parameters, for example text that have to be typed, timing relations, etc. Then we create an instance of the button and an associated click action. This model instance conforming to the input metamodel will be the input model of the requirement, expressing that the user types the login credentials and then clicks on login button.

We may express the expected outcome by instantiating a page load action (i.e., the output model), which again can hold the expected page title as parameter. Using our proposed representation, this pair of input and output model will express the requirement mentioned above.

For the example above, a straightforward action metamodel can be constructed using Selenium command set. Selenium is a browser automation tool, that is very popular for web application's test automation [3]. The command set (usually referred to as "selenese") contains actions in three category.

Actions provide functionality to manipulate the application, e.g., click on or select an item.

Accessors are used to identify and save the current state of the application.

Assertions can verify that the state of the application matches what expected, e.g., verify that a check box is checked, etc.

In case of a web-based software, using these as actions in the metamodel and every GUI element from the interface, we get a metamodel that is appropriate to describe almost every interactions (including timing information and any desired parameter) between the software and its environment.

IV. Use of context-related requirements for software testing

Software testing is the process of evaluating the quality of the software under test (SUT) by controlled execution, usually with the primary aim to reveal inadequate behaviour. In functional (i.e., black-box) software testing we treat software as it would be a function transforming the input data to the output data. We do not have or do not use the information about the internal structure of the SUT. This means that the tester has to be familiar with relations between the SUT and the environment. At this point we may exploit the proposed requirements representation approach, since it expresses how the SUT

interact with its environment without dealing with details or internal information. These can be used to derive test case descriptions for the SUT.

A possible way of test case generation based on the requirement models is model generation, i.e., model instance creation conforming to the given metamodel. In simple cases the input model of the requirements model can be used as an input test data directly, and we can compare the output model to the actual output data. However, when we need more complicated models to exercise the SUT (e.g., we have constraints that force the presence of particular elements), we may expand the initial input model by instantiating further elements from the metamodel, and get a model that contains other objects and actions.

In the previous example, we may instantiate every clickable GUI element from the metamodel and add a click event to all of these, and then type the login credentials and click on login button. With this extension we can verify that the requirement is satisfied even if the user previously clicked every element on the web interface.

In addition, we may automate the generation of such models. Search-based software engineering (SBSE) is the use of search-based optimization algorithms (usually metaheuristic search techniques) to software engineering problems. The key ingredients for the application of search-based optimization is the choice of representation of the solutions and the definition of the fitness function. In case of test generation on the basis of context related requirements, we can represent solutions with model instances, and we can compose the fitness function from the test goals and coverage metrics (e.g., minimum number of elements, instantiation of particular elements), that will guide the model generation. Thus we can use SBSE to search for model instances that are appropriate according to our selected conditions. A search-based model generation approach is presented in [4] and [5].

V. Conclusion and future work

In this paper we have briefly introduced a novel approach for high-level description of context-related functional requirements. We create a metamodel, that describes the context of the software, and contains every possible object that may appear in the context and every possible action that may affect the software or can be executed by the software. We use the context metamodel and define requirements as model instances conforming to this metamodel. We think that this new approach to requirement modelling is advantageous when generating functional test cases.

In the future we plan to implement a framework which is able to generate test cases based on the context-related requirement models and additional boundary conditions.

Acknowledgement

This work was partially funded by TÁMOP-4.2.1/B-09/1/KMR-2010-0002 and ARTEMIS-JU Grant Agreement 100233 (R3-COP).

References

[1] B. H. C. Cheng and J. M. Atlee, "Research directions in requirements engineering," in *2007 Future of Software Engineering*, FOSE '07, pp. 285–303, Washington, DC, USA, 2007. IEEE Computer Society, URL: http://dx.doi.org/10.1109/FOSE.2007.17.

[2] T. Cohene and S. Easterbrook, "Contextual risk analysis for interview design," in *Proceedings of the 13th IEEE International Conference on Requirements Engineering*, pp. 95–104, 2005.

[3] Selenium Project, *Selenium Documentation*, 2011, URL: http://seleniumhq.org/docs/.

[4] Z. Szatmári, J. Oláh, and I. Majzik, "Ontology-based test data generation using metaheuristics," in *Proc. of the 8th International Conference on Informatics in Control, Automation and Robotics*, 2011.

[5] J. Oláh and I. Majzik, "Search-based functional test data generation using data metamodel," in *Proc. of the 3rd International Symposium on Searc Based Software Engineering*, 2011.

FORWARD SATURATION BASED MODEL CHECKING

András VÖRÖS
Advisor: Tamás BARTHA

I. Introduction

Formal methods are gaining importance as safety critical, distributed and embedded systems are becoming widespread. By using formal verification we can find errors or we can prove the correctness in an early stage of the design. Model checking is an automatic verification method to check discrete, finite state models. In the last 20 years many model checking algorithms appeared: in this paper we focus on one of them, the so-called *saturation* algorithm. Saturation is a symbolic state space generation and model checking algorithm, which is efficient for globally asynchronous, locally synchronous (GALS) models. Former work presented structural model checking algorithms using saturation and constrained saturation, which were based on the classical backward traversal of the state space. In this paper we introduce saturation model checking based on forward state traversal. We hope this research direction to further improve the efficiency of model checking algorithms.

II. Preliminaries

Petri nets [1] are graphical models for concurrent and asynchronous systems, providing both structural and dynamical analysis.

An *event* in the system is the firing of an enabled transition. The firing of transitions is non-deterministic. The *state space* of a Petri net is the set of states reachable through transition firings.

In order to examine a model (for example a Petri net), we have to explore its possible dynamic behaviour, i.e. the state space. Traditional *symbolic state space exploration* uses encoding for the traversed state space, and stores this compact, encoded representation only. Decision diagrams proved to be an efficient form of symbolic storage, as applied reduction rules provide a compact representation form. Another important advantage is that symbolic methods enable us to manipulate large set of states efficiently.

A *Multiple-valued Decision Diagram* (MDD) is a directed acyclic graph, representing a function f consisting of K variables: $f : \{0, 1, \ldots\}^K \to \{0, 1\}$. An MDD has a node set containing two types of nodes: non-terminal and two terminal nodes (0 and 1). The nodes are ordered into $K + 1$ levels. A non-terminal node is labelled by a variable index $0 < k \leq K$, indicating which level the node belongs to (which variable it represents), and has n_k (the domain size of the variable, in binary case $n_k = 2$) edges pointing to nodes in level $k - 1$ (the i-th edge of node n is written as $n[i]$). A terminal node is labelled by the variable index 0. Further information can be found in [2].

With the help of MDD based symbolic representation we are able to explore the state space of complex systems. The first step of symbolic state space generation is to encode the possible states. Traditional approach encodes each state with a certain variable assignment of state variables $(v_1, v_2 \ldots v_n)$, and stores it in a decision diagram. To encode the possible state changes, we have to encode the transition relation, the so called *next-state* function. This can be done in a $2n$ level decision diagram with variables: $\mathcal{N} = (v_1, v_2 \ldots v_n, v'_1, v'_2 \ldots v'_n)$, where the first n variables represent the *"from"*, and second n variables the *"to"* states. The next-state function represents the possibly reachable states in one step.

Usually the state space traversal builds the next-state relation using a breadth first search. The reachable set of states S from a given initial state s_0 is the *transitive closure* (in other words: the *fixed-point*)

of the next-state relation: $S = \mathcal{N}^*(s_o)$. Saturation based state space exploration differs from traditional methods as it combines symbolic methods with a special iteration strategy. This strategy is proved to be very efficient for asynchronous systems modelled with Petri nets.

The saturation algorithm consists of the following steps:

- *Decomposition:* Petri nets can be decomposed into local submodels. The global state is the composition of the components' local states: $s_g = (s_1, s_2, \ldots, s_n)$, where n is the number of components, and s_n is the local state of the n-th component. This decomposition is the first step of the saturation algorithm.
- *Event localization:* As the effects of the transitions are usually local to the component they belong to, we can omit these events from other sub-models, which makes the state space traversal more efficient. For each event e we set the *border of its effect* by the top (top_e) and bottom (bot_e) levels (submodels). Outside of this interval we omit the event e from the exploration.
- *Special iteration strategy:* Saturation iterates through the MDD nodes and generates the whole state space representation using a node-to-node transitive closure. In this way saturation avoids the peak size of the MDD to be much larger than the final size, which is a critical problem in traditional approaches. Let $\mathcal{B}(k, p)$ represent the set of states represented by the MDD rooted at node p, at level k. Saturation applies $\mathcal{N}^*$ locally to the nodes from the bottom of the MDD to the top. Let $\mathcal{E}$ be the set of events affecting the k-th level and below, so $top_e \leq k$. We call a node p at level k saturated, iff node $\mathcal{B}(k, p) = \bigcup_{\forall e \in \mathcal{E}} \mathcal{N}_e^*(\mathcal{B}(k, p))$. The state space generation ends when the node at the top level becomes saturated, so it represents the state space: $S = \mathcal{N}^*(s_0)$.
- *Encoding of the next-state function:* Saturation algorithm uses a disjunctive-conjunctive transition relation decomposition [3], where the global next state relation $\mathcal{N}$ is constructed as the disjunction of the transition relations for all event: $\mathcal{N} = \bigcup_{\forall e} \mathcal{N}_e$. Each transition relation $\mathcal{N}_e$ is then constructed as the conjunction of sub-relations. Sub-relations are constructed model dependently, as the chosen higher level model determines how they can be efficiently created. In the case of ordinary Petri nets conjunctive representation can be based on *Kronecker matrices* [2], however for scalability and flexibility reasons we employ symbolic representation of the Next-state functions.
- *Building the MDD representation of the state space:* At first we build the MDD representing the initial state. Then we start to saturate the nodes in the first level by trying to fire all e events where $top_e = 1$. After finishing the first level, we saturate all nodes at the second level by firing all events, where $top_e = 2$. If new nodes are created at the first level by the firing, they are also saturated recursively. The procedure is continued at every level k for events, where $top_e = k$. When new nodes are created in a level below the current one, they are also recursively saturated. If the root node at the top level is saturated, the algorithm terminates. Now the MDD represents the whole state space with the next-state relation encoded in Kronecker matrices or symbolically in MDD-s.
- *State space representation as an MDD:* A level of the MDD generated during saturation represents the local state space of a submodel. The possible states of the submodel constitute the domain of the variables in the MDD. Each local state space is encoded in a variable.

Model checking [4] is an automatic technique for verifying finite state systems. Given a model defined for example as a Petri net in our context, model checking decides whether the model fulfils the specification. Formally: let M be a Kripke structure (i.e. state–transition graph). Let f be a formula of temporal logic (i.e. the specification). The goal of model checking is to find all states s of M such that $M, s \models f$ ("$\models$" means "satisfies").

State space generation serves as a prerequisite for the structural model checking: verifying temporal properties needs the state space and transition relation representation. *CTL* (Computation Tree Logic) is widely used to express temporal specifications of systems, as it has expressive syntax and there are

efficient algorithms for its analysis. Operators occur in pairs in CTL: the *path quantifier*, either A (on all paths) or E (there exists a path), is followed by the *tense operator*, one of X (next), F (future, or finally), G (globally), and U (until). However we only need to implement 3 of the 8 possible pairings due to the duality [4]: EX, EU, EG, and we can express the remaining with the help of them in the following way: $\mathsf{EF}p \equiv \mathsf{E}[true\ \mathsf{U}p]$, $\mathsf{AX}p \equiv \neg\mathsf{EX}\neg p$, $\mathsf{AG}p \equiv \neg\mathsf{EF}\neg p$, $\mathsf{AF}p \equiv \neg\mathsf{EG}\neg p$, $\mathsf{A}[p\ \mathsf{U}\ q] \equiv \neg\mathsf{E}[\neg q\ \mathsf{U}\ (\neg p \wedge \neg q)] \wedge \neg\mathsf{EG}\neg q$.

The CTL model checking algorithm efficiently utilizes the data structures created previously, during the state space exploration. As CTL operators express next-state relations and fixed point properties, we have to efficiently express the inverse of the next-state function $\mathcal{N}^{-1}$. The semantics of the 3 CTL operators:

- **EX**: $i^0 \vDash \mathsf{EX}p$ iff $\exists i^1 \in \mathcal{N}(i^0)$ s.t. $i^1 \vDash p$. This means that EX corresponds to the inverse $\mathcal{N}$ function, applying one step backward through the next-state relation, formally: $\mathsf{EX}\,p = \mathcal{N}^{-1}(p)$
- **EG**: $i^0 \vDash \mathsf{EG}p$ iff $\forall n \geq 0, \exists i^n \in \mathcal{N}(i^{n-1})$ s.t. $i^n \vDash p$ so that there is a strongly connected component containing states satisfying p. This computation needs a greatest fixed-point computation: $\mathsf{EG}p$ = **gfp** $Z[p \wedge \mathsf{EX}\,Z]$
- **EU**: $i^0 \vDash \mathsf{E}[p\ \mathsf{U}\ q]$ iff $\exists n \geq 0, \exists i^1 \in \mathcal{N}(i^0), \ldots, \exists i^n \in \mathcal{N}(i^{n-1})$ s.t. $i^n \vDash q$ and $i^m \vDash p$ for all $m < n$. Informally: we search for a state q reached through only states satisfying p. The computation of this property needs a least fixed-point computation: $\mathsf{E}[p\ \mathsf{U}\ q]$ = **lfp** $Z[q \vee (p \wedge \mathsf{EX}\,Z)]$

As it is easy to see, these operations and fixed-point calculations are based on the *pre-image* (inverse Next-state) computation operator: $\mathcal{N}^{-1}$. The question comes naturally: can we replace backward traversal based operators? The idea of forward state traversal symbolic model checking appeared to replace backward state traversal.

In order to be able to do forward model checking, we have to convert the semantics of the backward model checking [5]. If we examine a model with initial state s_0, where exactly predicate p_0 is true, so that $s_0 \vDash f$ can be rewritten: $s_0 \vDash f \iff p_0 \wedge f \neq \mathit{false} \iff p_0 \wedge \neg f = \mathit{false}$. The semantic of forward and backward traversal model checking differs, like the expressible possible properties [6]. Forward model checking is built on path expressions, which is contrary to the approach used by backward structural model checking. The forward model checking approach builds a so-called *path set expression (PSE)* and checks its validity in the model [7]. The basic elements of PSE-s are the following (p and q are propositional formulas):

- $[p]$ matches every one-step sequence, which satisfies p
- $[p]^*[q]$ matches every finite sequence, which ends in a state satisfying q, and all states before satisfies p. This is the forward traversal counterpart of the $\mathsf{E}[p\mathsf{U}q]$ CTL operator.
- $[p]^\omega$ matches every infinite sequence such that each state satisfies p. This is the forward traversal counterpart of the $\mathsf{EG}\ p$ CTL operator.
- $\alpha\beta$ matches to every sequence, such that the first finite part matches formula α, and after it the (tail) sequence matches β
- $\alpha : \beta$ matches to every sequence, such that the first finite part matches formula α, then there is a state, where both α and β is true, then the last (tail) sequence matches β

The main forward traversal evaluation procedures and their semantics are the following:

- $\mathsf{fw}([p][f])$: it computes the PSE $[p][f]$, the procedure computes $\mathcal{N}(p) \wedge f \neq false$,
- $\mathsf{fwU}(p, q)$: it computes the PSE $[p] : [q]^*$, the procedure computes the forward least-fixed point **lfp** $Z[p \vee \mathcal{N}(Z \wedge q)]$
- $\mathsf{fwG}(init, p)$: it computes the PSE $[init] : [p]^\omega$, the procedure computes the forward greatest fixed-pont **gfp** $Z[p \wedge \mathcal{N}(Z)]$; to be able to compute this greatest fixed-point, we have to compute the reachable states from $init$ state satisfying p: **lfp** $Z[init \vee \mathcal{N}(Z \wedge p)]$, which can be done with the formerly defined procedure: $\mathsf{fwU}(init, p)$

III. Saturation based forward model checking

Our aim is to apply saturation in forward model checking, so we need saturation based fw, fwU and fwG procedures. At the end of this section we show how CTL expressions are computable with these forward traversal procedures.

For this reason, we employ the so-called constrained saturation algorithm [8], with small modifications. The traditional constrained saturation algorithm uses $\mathcal{N}_l^{-1}$ to explore the states, so it uses backward state traversal. By changing the direction, and using $\mathcal{N}_l$ in the algorithm $ConsSaturate(p, q)$ [8], it will compute exactly the same states as procedure $\mathsf{fwU}(p, q)$. The main advantage of this approach is that we can avoid the intersection operations which are applied in the traditional approaches at every breadth-first step. So we have an algorithm to compute $\mathsf{fwU}(p, q)$. We have to be able to compute $\mathsf{fw}([p][f])$ and $\mathsf{fwG}(p)$ too to be able to handle more CTL operators. $\mathsf{fw}([p][f])$ needs a Next-state computation, which does not use saturation, it can be done with former approaches. The algorithm computing $\mathsf{fwG}(init, p)$ starts with computing $ConsSaturate(init, q)$, and then by standard breadth first search it computes the greatest fixed-point **gfp** $Z[p \wedge \mathcal{N}(Z)]$.

In the following the basic CTL expressions and their forward traversal based counterparts are:

- Forward EX evaluation: It can be changed to $\mathsf{fw}([p][f])$. So we can replace the outermost EX evaluation with Next-state computation, $\mathsf{fw}([p][f]) : p \wedge \mathsf{EX}f \neq \mathit{false} \Longleftrightarrow \mathcal{N}(p) \wedge f \neq \mathit{false}$
- Forward EU evaluation: It can be changed to $\mathsf{fwU}(p, q)$, so we can replace the outermost EU evaluation as follows: $p \wedge \mathsf{E}[q \ \ \mathsf{U} \ \ f] \neq \mathit{false} \Longleftrightarrow \mathsf{fwU}(p, q) \wedge f \neq \mathit{false}$
- Forward EG evaluation: Using the fwG operator, we can replace the outermost EG evaluation as follows: $p \wedge \mathsf{EG}\ q \neq \mathit{false} \Longleftrightarrow \mathsf{fwG}(p, q) \neq \mathit{false}$

This way many of the CTL expressions are convertible to model check in a forward manner. However, there are still some examples, which are not, according to the literature [6]. The theoretical question is still open: which formulas are expressible with this logic [6].

IV. Conclusion

In this paper I have presented how forward traversal based CTL model checking can be done with the help of the saturation algorithm. The main motivation of this work is to further improve the efficiency of saturation by avoiding the full state space exploration which was necessary with former saturation based model checking algorithms.

In the future we plan to implement these algorithms and we would like to further improve EG computation, which seems to be the bottleneck of our approach.

References

[1] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, 77(4):541–580, April 1989.

[2] G. Ciardo and A. S. Miner, "Storage alternatives for large structured state spaces," in *Proceedings of the 9th ICCPE: Modelling Techniques and Tools*, pp. 44–57, London, UK, 1997. Springer-Verlag.

[3] G. Ciardo and A. Yu, "Saturation-based symbolic reachability analysis using conjunctive and disjunctive partitioning," *Correct Hardware Design and Verification Methods*, 3725:146–161, 2005.

[4] E. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*, The MIT Press, 1999.

[5] H. Iwashita, T. Nakata, and F. Hirose, "CTL model checking based on forward state traversal," in *1996 IEEE/ACM International Conference on Computer-Aided Design*, pp. 82–87. IEEE, 1996.

[6] T. Henzinger, O. Kupferman, and S. Qadeer, "From pre-historic to post-modern symbolic model checking," in *Computer Aided Verification*, pp. 195–206. Springer, 1998.

[7] H. Iwashita and T. Nakata, "Forward model checking techniques oriented to buggy designs," *Proceedings of IEEE International Conference on Computer Aided Design (ICCAD) ICCAD-97*, pp. 400–404, 1997.

[8] Y. Zhao and G. Ciardo, "Symbolic CTL model checking of asynchronous systems using constrained saturation," in *Proceedings of the 7th International Symposium on Automated Technology for Verification and Analysis*, ATVA '09, pp. 368–381, Berlin, Heidelberg, 2009. Springer-Verlag.

FOUR PARAMETER SINE WAVE ESTIMATION IN FREQUENCY DOMAIN

Vilmos PÁLFI
Advisor: István KOLLÁR

I. Introduction

Accurate characterization of analog-to-digital converters (ADC) is an important field of measurement technology. A widely used method for characterizing ADC's is the so called histogram test [1]. In this method first the ADC is excited with a sine wave input, then the integral nonlinearity (INL) and differential nonlinearity (DNL) can be identified from the number of samples in the code bins. Obviously, the parameters of the applied sine wave and the sampling frequency are important. The IEEE standard for ADC testing [2] defines that sampling should be coherent and the number of periods has to be relative prime to the number of samples, in order to achieve the most accurate results. However, the satisfaction of these conditions can only be checked from the measured signal, since both the frequency of the signal generator and of sampling have limited precision. With the increase of the number of samples to obtain more precise measurement of the characteristics of the device under test, even a small deviation from coherence in the sampling causes significant errors in the characterization with false INL and DNL values, and also an increase in the time required to execute the least-squares four parameter sine wave fit algorithm. In this paper an increased-speed algorithm is presented with the capability to decide if the test signal is suitable to test the ADC or not. In some cases the coherence condition can be assured with further preprocessing steps, this will also be studied.
[1]

II. Background

A. Histogram Test

In the histogram test the ADC is excited with a sine wave input. The form of the input is

$$\begin{aligned} x[k] = A_1 \sin(\omega k + \varphi) + C = \\ A\cos(\omega k) + Bsin(\omega k) + C, \\ k = 1, 2, \ldots N. \end{aligned} \tag{1}$$

Where $A_1 = \sqrt{A^2 + B^2}$ is the amplitude, ω is the angular frequency, C is the DC offset, and $\varphi = \arctan(B, A)$ is the initial phase, and N is the number of samples. Let $H[i]$ be the total number of samples received in the code bin i, and

$$H_c[k] = \sum_{i=0}^{k} H[i]. \tag{2}$$

If we know the parameters A_1, and C (A, B, and C), then the nth transition level can be estimated as

$$T[n] = C - A_1 \cos(\pi H_c[n-1]/N). \tag{3}$$

The kth code bin width is given by

$$W[k] = T[k+1] - T[k], \tag{4}$$

and the INL and DNL values can be estimated.

[1] This work is based on "Verification of parameter settings in ADC test with sine fitting", IMTC 2012 abstract

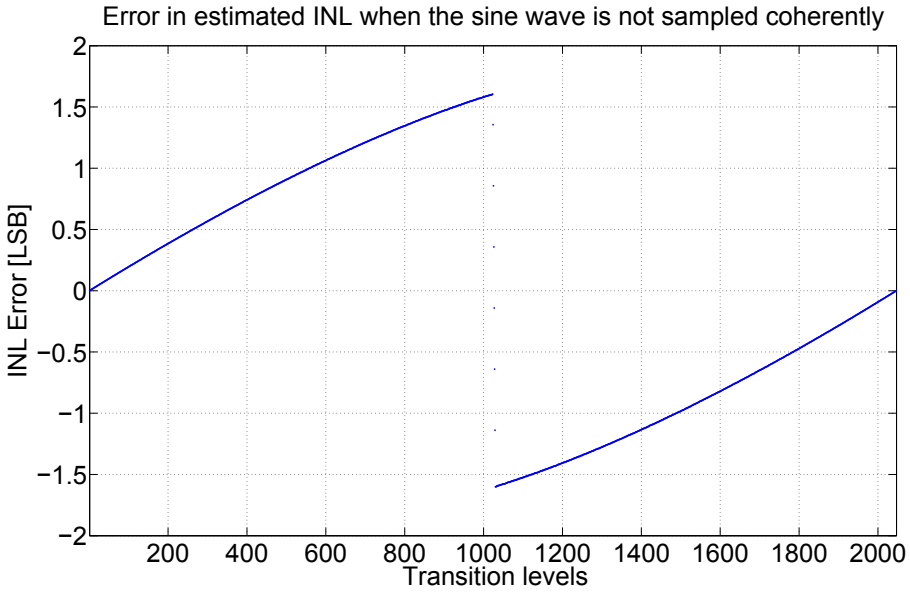


Figure 1: Errors in the INL estimation when not coherently sampled sine wave is applied

B. Standards for choosing signal and sampling frequencies, record length and periods per record

The sine wave input is optimal if the sampling is coherent, so an integer number of periods are sampled, and all the samples represent different phases. If the sampling frequency is f_s, the number of samples is N, then the signal frequency is optimal if

$$f_x = \frac{J}{N} f_s, \tag{5}$$

where J is an integer relative prime to N, and f_x is the signal frequency. Fig. 1 shows why it is important to precisely meet the coherency condition. An ideal ADC was tested with one period of a sine wave, the record length was $N = 2^{20} = 1048576$. To have one period, the optimal signal frequency would be $f_{opt} = f_s/N$, where f_s is the sampling frequency. The applied signal frequency was $f_x = 1.001 f_{opt}$. As Fig. 1 shows, this difference causes significant errors in estimating the transition levels and then in calculating the INL. The second condition ensures that every input sample represents a different phase, with uniform phase distribution. For example, if both the number of samples and number of periods can be divided by 2, then the first and the second half of the sequence are exactly the same (apart from the noise), so the number of useful samples is half the number of total samples.

III. FOUR-PARAMETER SINE WAVE ESTIMATION IN THE FREQUENCY DOMAIN

A. Implementation properties

The goal was to create an algorithm which can identify the parameters of the sine wave much faster than the ordinary least squares method. The sine wave's four parameter form is

$$x(k) = A\cos(2\pi k f) + B\sin(2\pi k f) + C. \tag{6}$$

This form is very advantageous because it is linear in 3 parameters. Due to the nonlinearity in frequency, an iterative numerical method is needed to find the minimum of the least squares cost function as a solution in closed form is not available for nonlinear LS. The output of the ADC is windowed with samples of the four-term Blackman-Harris window, which has side lobes under -91 dB. Then FFT is

Table 1: "RESULTS FOR "REAL-LIKE" QUANTIZER"

Estimator	NoB	Added noise	
	Added noise	0	$[-q, q]$
time	μ	$-3.09 \cdot 10^{-8}$	$7.22 \cdot 10^{-8}$
	σ	$6.70 \cdot 10^{-7}$	$2.20 \cdot 10^{-6}$
freq	μ	$-8.28 \cdot 10^{-8}$	$3.41 \cdot 10^{-9}$
	σ	$7.58 \cdot 10^{-7}$	$2.37 \cdot 10^{-6}$

performed, and the cost function is calculated in the frequency domain. After calculating the initial values of the parameters, Gauss-Newton algorithm is used to find the minimum. Since the Blackman-Harris window has small side lobes, the information in the frequency domain is compressed into a few points. In the fitting algorithm we use 30 points to determine the parameters, independently from the length of the time domain signal. The original, time domain algorithm uses every time domain sample in the fit, so for large records the computational burden is high. In this frequency domain fit only the evaluation speed of the FFT depends on N, so it is much faster thus it allows the use of large records.

B. Properties of the estimator

In this section the statistical properties of the estimator will be discussed. We assume that the noise in the frequency domain is approximately normally distributed. This assumption is based on the Central Limit theorem. The noise at the FFT output will be Gaussian independently from its original distribution. The larger the number of samples, the previous approximation is truer. This means that the weighted least squares estimation used is a maximum likelihood estimation, so it has a number of advantageous properties:

- the estimator is asymptotically normally distributed,
- the estimator is asymptotically unbiased,
- the covariance matrix of the estimator asymptotically reaches the Cramer-Rao lower bound.

Due to the first property the estimators can be fully characterized with their mean value (μ) and standard deviation (σ). In the next section the estimator is compared to the result of the ordinary least squares method, described in the standard [2].

C. Comparing the algorithm to the original estimation

In the following tests the standard deviations and the mean values of the estimators were compared between time domain and frequency domain to illustrate the quality of the frequency domain fit of the Blackman-Harris weighted data. In these tests the amplitude, phase, offset and frequency were chosen as random variables uniformly distributed in $[0.5, 1]$, $[0, 2\pi]$, $[-0.5, 0.5]$, $[4f_s/N, 10f_s/N]$, respectively, and the record length was $N = 2^{20}$. To simulate life-like circumstances, a quantizer based on real nonlinear characteristics was created. The number of fraction bits was 10, the full scale range (FS) was 1 V, and the quantizer was bipolar so it worked between -1 V and 1 V, thus it had a sign bit. Fig. 2 shows the INL and DNL characteristics of the quantizer. First we tested with pure sine wave, then the input was disturbed with uniformly distributed noise in $[-q, q]$. The results are shown in Table 1.

IV. Using the estimator to preprocess data

Two important conditions should be met to obtain the best results. These are coherent sampling and the appropriate relation between the number of samples and the number of periods of the sine wave

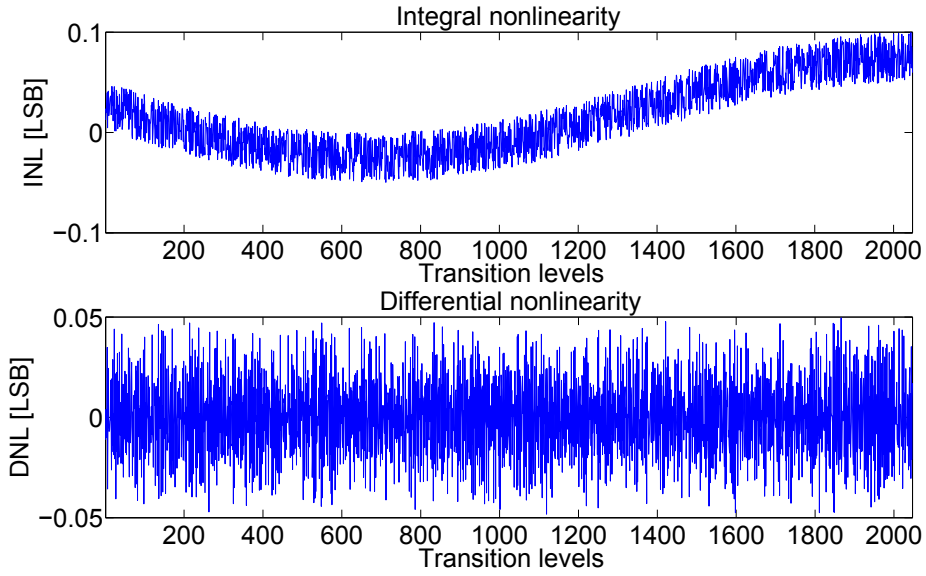


Figure 2: INL and DNL characteristics of the quantizer

applied. Both of the conditions can be met if the input signal's frequency is chosen correctly. So the frequency domain estimator is a feasible tool to check if the input sine wave is appropriate to test the converter. The distribution of the phases depends on the deviation of the wave frequency from coherence. In details, if

$$\frac{f_x}{f_s} = \frac{J}{N} + \Delta \tag{7}$$

then none of the sampling points deviate from the ideal phase from more than $\frac{2\pi}{4N}$ (where $\frac{2\pi}{N}$ is the ideal distance between two adjacent sampling points) if

$$|\Delta| \leq \frac{1}{2JN} \tag{8}$$

is true [3]. So we can check if the input sine wave meets the previous conditions or not, and warn the user. Furthermore, in the case when the sampling is almost coherent, but we have a few additional samples from the beginning of the next period, we can calculate how many samples should be thrown away to reach the best results. If a few samples are missing from the end of the whole last period, we can either throw away the whole period or suggest new sampling.

V. Conclusion

The main advantage of the estimator that it can determine in no time (compared to the original least squares) if the measured signal is applicable to characterize an ADC properly, and in some special cases it can modify the input to ensure better results.

References

[1] J. Blair, "Histogram measurement of ADC nonlinearities using sine waves," *IEEE Transactions on Instrumentation and Measurement*, 43(3):373–383, Jun 1994.

[2] "IEEE standard for terminology and test methods for analog-to-digital converters," *IEEE Std 1241-2010 (Revision of IEEE Std 1241-2000)*, pp. 1 –139, 14 2011.

[3] P. Carbone and G. Chiorboli, "Adc sinewave histogram testing with quasi-coherent sampling," in *Instrumentation and Measurement Technology Conference, 2000. IMTC 2000. Proceedings of the 17th IEEE*, vol. 1, pp. 108 –113 vol.1, 2000.

FINITE-DIFFERENCE SIMULATION OF ACOUSTIC WAVE PROPAGATION IN ENCLOSURES

Róbert GALAMBOS
Advisors: László SUJBERT, Balázs BANK

I. Introduction

There are various cases where the acoustic behavior of an enclosure must be calculated. Such a problem is the placement of loudspeakers in a given space to achieve the best sound experience. The ray tracing is a well known numerical technique but mostly suitable for high frequency analysis where geometric approximations can be applied [1]. For very low frequencies and small enclosures, modal approach based on modal profiles calculated in the frequency domain, by finite-element method is appropriate in many cases [2]. In the middle-frequency range, where the wave behavior is still dominant, and large number of modes are important, the simulation of the acoustic space is hard to treat. With the increasing calculation capacity of the computers the time-domain approaches seem to be appropriate for solving the problem.

This paper demonstrates the simulation of small spaces with different geometries and wall behaviors, by the finite-difference time-domain (FDTD) method [2, 3, 4]. It can be utilized for speaker configuration relevant compensation filter design, and for simulation of various multirate signal processing applications.

II. The FDTD Simulation Method

The finite-difference time-domain method utilizes two coupled first-order differential equations. It is a finite-difference approximation of the time and space derivatives of the wave equation [5, 6].

The first equation of the two coupled differential equations is the linear force equation, that is valid for acoustic process with small amplitude where the pressure p and the particle velocity u are related in the following way:

$$\nabla p = -\rho_0 \frac{\partial \mathbf{u}}{\partial t} \tag{1}$$

where ρ_0 is the density of the transmission medium in $\mathrm{kg/m^3}$.

The second equation of the two coupled differential equations is the linear continuity equation:

$$\nabla \cdot \mathbf{u} = -\frac{1}{c^2 \rho_0} \frac{\partial p}{\partial t} \tag{2}$$

where c is the wave propagation speed in the medium in $\mathrm{m/s}$.

The most common discretisation is carried out on a rectangular grid where pressure and particle velocity are the unknown quantities. The grid is basically a double grid where the pressure points are equidistantly sampled in all the three directions, and between each two pressure points there is a velocity point in the middle. The boundaries are velocity points. This is displayed in Figure 1.

This means that pressure points are sampled at every $(x; y; z)$ where $x = \delta x \cdot h; y = \delta y \cdot h; z = \delta z \cdot h$ and the time is sampled at $t = \delta t \cdot k$ where $h \in \mathbb{Z}$ and $k \in \mathbb{Z}$. (1) and (2) can be sampled in time and space using the sample rate $1/k$ Hz and $1/h$ $\mathrm{m^{-1}}$.

From (1) the velocity can be determined at the following positions (upper index shows only the relevant direction):

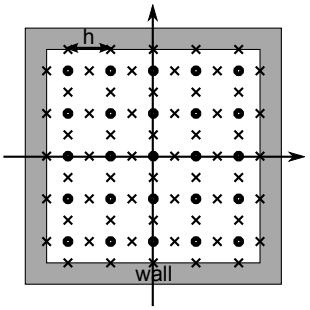


Figure 1: One slice of the grid system. X - marks the velocity and O - marks the pressure points

$$u^x\left(x \pm \frac{\delta x}{2}, y, z, t + \frac{\delta t}{2}\right), \quad u^y\left(x, y \pm \frac{\delta y}{2}, z, t + \frac{\delta t}{2}\right), \quad u^y\left(x, y, z \pm \frac{\delta z}{2}, t + \frac{\delta t}{2}\right) \tag{3}$$

This is computed at the time $t + \frac{\delta t}{2}$, and the velocity can be determined as follows:

$$\begin{aligned} u^x\left(x + \frac{\delta x}{2}, y, z, t + \frac{\delta t}{2}\right) &= u^x\left(x + \frac{\delta x}{2}, y, z, t - \frac{\delta t}{2}\right) - \frac{k}{h\rho_0}\left[p\left(x + \delta x, y, z, t\right) - p\left(x, y, z, t\right)\right] \\ u^y\left(x, y + \frac{\delta y}{2}, z, t + \frac{\delta t}{2}\right) &= u^y\left(x, y + \frac{\delta y}{2}, z, t - \frac{\delta t}{2}\right) - \frac{k}{h\rho_0}\left[p\left(x, y + \delta y, z, t\right) - p_{x,y,z}\left(t\right)\right] \\ u^z\left(x, y, z + \frac{\delta z}{2}, t + \frac{\delta t}{2}\right) &= u^z\left(x, y, z + \frac{\delta z}{2}, t - \frac{\delta t}{2}\right) - \frac{k}{h\rho_0}\left[p\left(x, y, z + \delta z, t\right) - p_{x,y,z}\left(t\right)\right] \end{aligned} \tag{4}$$

From (2) the acoustic pressure can be calculated as follows:

$$\begin{aligned} p\left(x, y, z, t + \delta t\right) =& p\left(x, y, z, t\right) - \frac{c^2\rho_0 k}{h}\left[u^x\left(x + \frac{\delta x}{2}, y, z, t + \frac{\delta t}{2}\right) - u^x\left(x - \frac{\delta x}{2}, y, z, t + \frac{\delta t}{2}\right)\right] \\ &- \frac{c^2\rho_0 k}{h}\left[u^y\left(x, y + \frac{\delta y}{2}, z, t + \frac{\delta t}{2}\right) - u^y\left(x, y - \frac{\delta y}{2}, z, t + \frac{\delta t}{2}\right)\right] \\ &- \frac{c^2\rho_0 k}{h}\left[u^z\left(x, y, z + \frac{\delta z}{2}, t + \frac{\delta t}{2}\right) - u^z\left(x, y, z - \frac{\delta z}{2}, t + \frac{\delta t}{2}\right)\right] \end{aligned} \tag{5}$$

There are several boundary conditions applicable for affecting the wave propagation in the simulation model.

A. Ideal wall simulation

Acoustic behavior of enclosures can be modeled by assuming ideally reflecting (lossless) walls. The initial value of the velocity grid is zero and it remains zero during the simulation. Therefore these grid points are overwritten with zero after the calculation of the velocity points.

B. Real wall simulation

Real walls are not lossless, and the simulations' accuracy can be improved by taking into account that the wall itself has an impedance that determines the absorption and the reflection of the wall.

Starting from (2) we take only one direction, the other directions can be derived from this:

$$u^x\left(x + \frac{\delta x}{2}, y, z, t + \frac{\delta t}{2}\right) = u^x\left(x + \frac{\delta x}{2}, y, z, t - \frac{\delta t}{2}\right) - \frac{k}{h\rho_0}\left[p\left(x + \delta x, y, z, t\right) - p\left(x, y, z, t\right)\right] \tag{6}$$

where the wall is located at the $\left(x+\frac{\delta x}{2}, y, z\right)$ coordinates. This means that we do not have any pressure information in the wall, therefore we can not use the $p\left(x, y, z, t\right)$. To calculate the derivative we can use an asymmetric finite-difference approximation:

$$p\left(x+\delta x, y, z, t\right)-p\left(x, y, z, t\right) \approx 2\left[p\left(x+\delta x, y, z, t\right)-p\left(x+\frac{\delta x}{2}, y, z, t\right)\right] \tag{7}$$

but we have no pressure grid point at the $\left(x+\frac{\delta x}{2}, y, z\right)$ coordinates. It can be calculated in the following way [2]:

$$p\left(x+\frac{\delta x}{2}, y, z, t\right)=Zu^{x}\left(x+\frac{\delta x}{2}, y, z, t\right) \tag{8}$$

where Z is the impedance of the wall. Another approximation must be applied to get the $u^{x}\left(x+\frac{\delta x}{2}, y, z, t\right)$ because we only have the $u^{x}\left(x+\frac{\delta x}{2}, y, z, t+\frac{\delta t}{2}\right)$ and the $u^{x}\left(x+\frac{\delta x}{2}, y, z, t-\frac{\delta t}{2}\right)$. There we use interpolation in the time domain as follows:

$$u^{x}\left(x+\frac{\delta x}{2}, y, z, t\right) \approx \frac{u^{x}\left(x+\frac{\delta x}{2}, y, z, t+\frac{\delta t}{2}\right)+u^{x}\left(x+\frac{\delta x}{2}, y, z, t-\frac{\delta t}{2}\right)}{2} \tag{9}$$

Applying these approximations on (6) we get the following formula [2, 4]:

$$\begin{aligned} u^{x}\left(x+\frac{\delta x}{2}, y, z, t+\frac{\delta t}{2}\right) =& \frac{\rho_{0} h / k-Z}{\rho_{0} h / k+Z} u^{x}\left(x+\frac{\delta x}{2}, y, z, t-\frac{\delta t}{2}\right) \\ & -\frac{2}{\rho_{0} h / k+Z} p\left(x+\delta x, y, z, t\right) \end{aligned} \tag{10}$$

To simplify the simulation model only the real part of the wall impedance is used. The impedance can be calculated from the absorption coefficient of the wall as follows [2]:

$$Z=\rho_{0} c \frac{1+\sqrt{1-\alpha}}{1-\sqrt{1-\alpha}} \tag{11}$$

where α is the absorption coefficient. The advantage of the above mentioned approximation is that only the nearest pressure points and the wall impedance is required to calculate the new velocities from the old values.

C. Stability

Because time and space equations are not independent from each other, choosing wrong sampling frequency can make the simulation unstable. To ensure the stability of the simulation (5) and (4) must satisfy the following [2, 3, 4, 7]:

$$c \cdot \delta t \leq \frac{1}{\sqrt{\frac{1}{\delta x^{2}}+\frac{1}{\delta y^{2}}+\frac{1}{\delta z^{2}}}} \tag{12}$$

As (12) estimates the maximum time step allowed from system stability point of view, it can be used to maximize the computational efficiency.

III. Implementation

The simulation was implemented in MATLAB. The system has been modeled using 4 rectangular grids. One for the pressure that has $m \times n \times p$ size, and one for each velocity direction with size increased by one in the given direction (eg. for x direction $(m+1) \times n \times p$). As the pressure has been

calculated at a certain time instant, the velocity can be calculated separately for all directions. It can be done because of the orthogonality. This results in a simple grid addressing, and matrix handling in the implementation.

The room can be defined with a 3D matrix, where each cell represents a pressure point. If a cell has a value of 0 the pressure point is handled as space, and if the cell has another value, the value will be considered to be the Z impedance of the wall, and reflection is calculated according to the wall impedance. This gives us the possibility to define a room with different wall behaviors. Figure 2 shows a small rectangular room slice from the top with a pillar in it. The sound source was placed near the upper right corner. The waves reflected from the walls and the pillar can clearly be seen in the figure. Behind the pillar the wavefront closes due to diffraction of the wave.

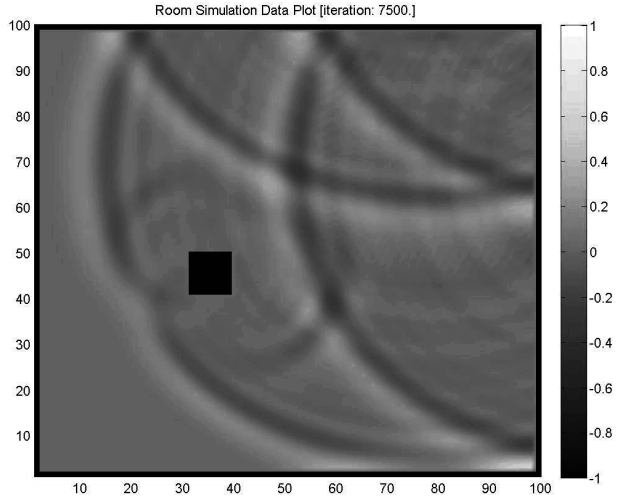


Figure 2: Pressure propagation in the simulation space (the pillar is displayed as a black rectangle)

IV. Conclusion

The presented simulation method is well suitable for low and middle frequency simulations with decent grid size. With higher grid resolution (for high frequency simulation) or bigger simulation space the computation complexity increases, making the method slow.

It is planned in the near future to add open space simulation to the implementation [7]. This will allow to simulate open and semi-open space, like the effect of an open door or window.

The MATLAB implementation is freely available at http://mit.bme.hu/˜galambos.

References

[1] M. Long, Ed., *Architectural Acoustics*, Elsevier Academic Press, 2006.

[2] S. B. N. Adrian Celestinos, "Low-frequency loudspeaker-room simulation using finite differences in the time domain-part 1: Analysis," *J. Audio Eng. Soc*, 56(10):772–786, 2008.

[3] D. Botteldooren, "Finite-difference time-domain simulation of low-frequency room acoustic problems," *Journal of the Acoustical Society of America*, 98(6):3302–3308, 1995.

[4] M. O. J. Hill, Adam J.; Hawksford, "Visualization and analysis tools for low-frequency propagation in a generalized 3d acoustic space," *J. Audio Eng. Soc*, 59(5):321–337, 2011.

[5] L. L. Beranek, Ed., *Acoustics*, Acoustical Society of America, 1986.

[6] K. U. I. Philip M. Morse, Ed., *Theoretical Acoustics*, McGraw-Hill, 1968.

[7] H. T. Takatoshi Yokota, Shinichi Sakamoto, "Sound field simulation method by combining finite difference time domain calculation and multi-channel reproduction technique," *Acoustical Science And Technology*, 25(1):15–23, 2004.

THE INTELLIGENT GREENHOUSE (SUMMARY OF PHD WORK IN 2011)

Peter EREDICS
Advisor: Tadeusz P. DOBROWIECKI

I. The intelligent greenhouse concept

My research area is the physically embedded intelligent information systems, with emphasis on the usage of AI methods in the field of greenhouse control systems. The goal of the research is to create an intelligent greenhouse control system being able to overcome all limitations of the current control solutions to end up with lower operation cost and higher owner complacency.

The concept of the intelligent greenhouse is based on three major differences compared to traditional solutions. The first difference is the application of goals instead of control parameters. While traditional control requires the owner to select temperature levels for all possible actuator actions (e.g. "open the windows if the temperature is higher than 23 degrees"), the intelligent control should work on goals specified by the owner (e.g. "the plants feel comfortable in the temperature range between 10 and 25 degrees"). This is a much more natural way of setting specification which handles the greenhouse owner as a plant-expert instead of assuming him to be a control-expert.

The second difference is the application of predictive modeling. This way control actions can take place in advance to avoid unwanted future situations. Traditional control solutions are reactive, i.e. the traditional control waits until the given temperature limit is reached, and takes actions only after a parameter is out of the desired range. If it is possible for the intelligent control to predict future state of the greenhouse, the control actions can be done in advance resulting in higher control performance.

The last difference is the way intelligent control handles the actuators. In a traditional control system all actuators are directly controlled via their control parameters, resulting in a fully unsynchronized control. The missing synchronization can result in suboptimal decisions (e.g. active heating along with open windows), or in the worst case can cause oscillations. Intelligent control system can apply AI planning methods to generate several potential joint plans for all the actuators. This way the synchronization is ensured by the planning method itself. Plans can be evaluated based on the goals set by the owner, and the most goal conforming plan will be executed as best suiting the needs of the plants.

A Previous results

In the previous years I have developed a measurement and traditional control system for a real industrial greenhouse near Szombathely. The measurement system records temperature measurements with high spatial resolution from all 18 desks of the house and other strategically selected locations in and around the greenhouse. Along with the local measurements regional temperature data and freely available weather forecasts are also collected (see [1] for more).

The recorded thermal data served as a basis for the development of a thermal greenhouse model. This model is aimed to create temperature predictions, essential to the second innovation of the intelligent greenhouse described above. The first attempts to model the greenhouse with one monolith model failed, as the complexity of the system (the number of inputs and prediction outputs) were too high. Some components of the system could be easily separated and realized independently (such as the external local temperature prediction module – more in [2]), but most of the components were strongly related to the greenhouse.

In order to create a suitable thermal model of the greenhouse the model had to be decomposed. With the separation of the quasi-independent components of the system I have proposed modular

model decomposition into 6 modules, see [3]. Some of these modules were easy to implement, such as the heating system modeled with a mixture of experts approach with two neural networks for the warming and cooling state of the house (see [4]), while other attempts failed. The main reason of failures was the small number of training samples available. Although the measurement system collected a lot of measurements, the quality of this data, and the length of the training time series samples were not satisfactory.

B Cleaning the measurement data

The large spatial resolution of the measurement system of the greenhouse resulted in a degraded reliability of the collected data records as the large number of sensors led to many instrumentation errors. The modeling methods working with measurement data require full records with reliable data from all the measurement locations. If all partially missing or partially faulty measurement records were dropped, the number of the available records would be very low, and the continuous time series segments would be very short. In order to repair the measurement data the data cleaning system outlined in Fig. 1 was developed. The data cleaning system has two kinds of inputs: the measurements recorded locally by the data acquisition system of the greenhouse, and the measurements and forecasts available from the internet. Both have a dedicated data cleaning application for identifying missing or invalid input records and to restore them if possible.

Data recorded in the greenhouse can contain several kinds of errors: in case of a known malfunctioning sensor the measurement system enters an error code in place of the measurements. In case of temporary sensor defects the recorded value can be out of range, or can be distorted with a large noise component. Error codes and out of range measurements can be easily identified while detecting measurement noise needs a more sophisticated method based on the knowledge of previous recordings. The simplest way of data reparation is copying previous records. This can be applied for any values but only in case of missing sequences of limited length.

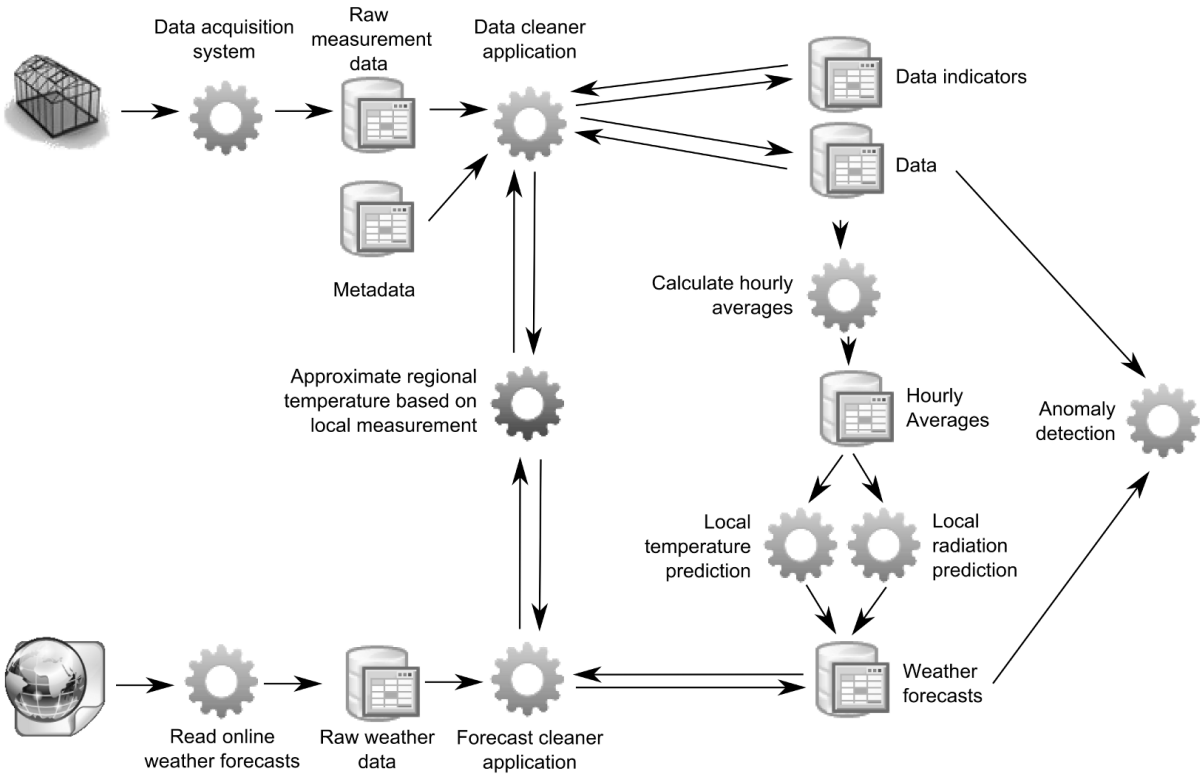


Figure 1: Structure of the data cleaning system

If the missing sequence is longer than a few steps (in case of the implemented system the data are collected every 5 minutes), the copy method fails. In this case a regression method can be applied using the last known measured value and optionally some other recorded values (typically radiation values). This regression method is also useful in case of missing online forecast data. If the internet connection is down the missing online prediction can be reproduced based on actual local measurements, available local forecast and radiation values.

Measurements from the desks inside the greenhouse are special, as recordings of the different desks are closely related to each other. This correlation can be used during the data reparation process, as missing desk measurements can be approximated by the spatial interpolation of known values from other desks around. Details of the data cleaning system were published in [5].

C Detecting anomalies

The data cleaning system was later extended with an additional module responsible for anomaly detection. The goal of anomaly detection is to identify unreliable sequences of the recorded data. Such sequences can be generated by the operation of the data cleaning system, or by a rapidly changing weather outside. Such fragments of the recorded data should be excluded from modeling in order to increase its performance and accuracy. The first step of anomaly detection was the extension of the recorded data vector with a large number of calculated attributes. These attributes were calculated as all possible reasonable differences between the various elements of the data vector. The second step is to create a system model based on a number of extended data vectors. This system model was represented by storing the mean value and the standard deviation for each attribute separately. The last step is the detection itself, when the new measurement vectors (after being extended) were compared to the system model: large deviations from the model were scored with a high anomaly score, and such records were labeled suspicious.

The anomaly detection method was tested on 10000 records collected in 2008. The method identified several real anomalies (data repaired poorly by the data cleaning system and also some rare weather phenomena), but it has to be noticed that the false alarm rate was also significant with the parameter setting used for testing. Results of the anomaly detection were published in [6].

D Thermal modeling of the greenhouse

With the repaired data vectors available a global greenhouse model was developed in cooperation with Kristóf Gáti. The aim of the model was to predict important temperature values for the inside zones of the greenhouse for 5-20 minutes ahead. We used CMAC (Cerebellar Model Articulation Controller) for the adaptive modeling with 5 dimensional inputs, and near 130000 data samples. The data was split into an 80000 samples long training section, and a 50000 long validation section. The results of the one step predictions can be seen in Fig. 2. The method yielded high error in case of the heating pipe temperature, but this is no surprise, as the pipe temperature is heavily determined by the heating control signal. The results were also poor in case of the external temperature prediction, but this has also secondary importance, as external forecasts are already available. The internal zone predictions have 0.5-1.6 degrees average absolute error, slowly rising with increasing prediction length (up to 0.8-1.8 degrees in case of the 4 steps prediction, i.e. 20 minutes time span). The results of the CMAC experiments were published in [7], however the precision of the predictions were not too high.

Following the CMAC experiments I started to experiment with other neural architectures in order to create more reliable predictions for the greenhouse. The goal was to create a simple MLP neural model for the temperature prediction of the most important zone of the greenhouse, namely the under shading screen temperature. These experiments failed so far as a very simple constant predictor (using the actual value of the selected parameter as the future prediction) was able to beat the neural model in all cases for a 5-30 minutes prediction. Later I switched to the zone above the shading screen, because it appeared to have less disturbing inter-zone relationships, but the results are still not yet acceptable, thus future work is needed.

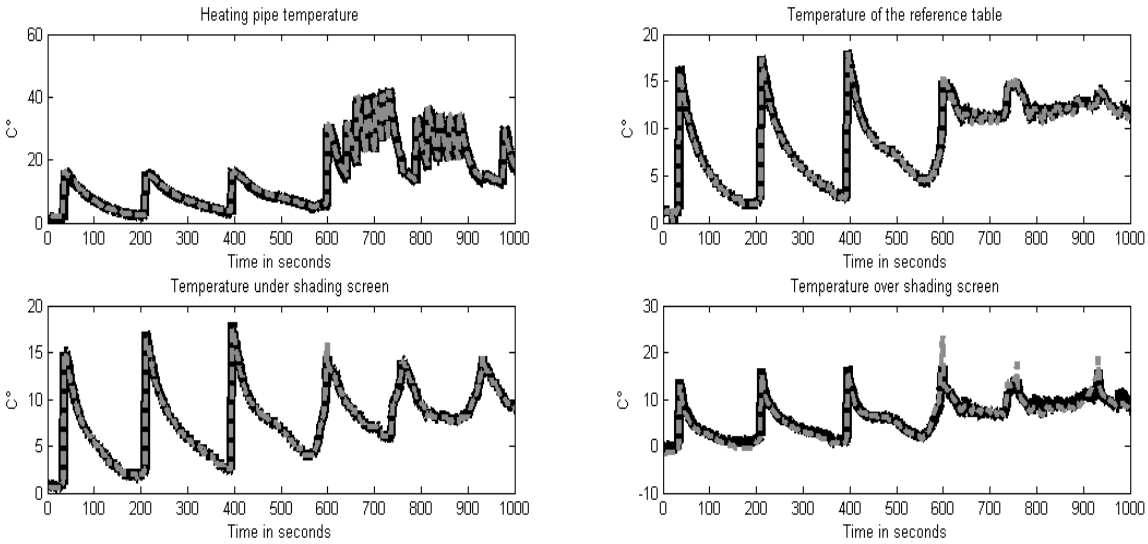


Figure 2: One step prediction for the temperature signals (predicted = black; actual = gray).

II. Other projects

Last year I have notable contribution to two projects at the department. The first project called Humeda was aimed to develop a decision support system for emergency physicians and emergency departments. The doctors were provided with a user interface to select the symptoms of the patient and the results of the examinations, and the system had to be able to list the possible diagnoses ordered by their relevance. My contribution to the project was the development and implementation of the inference engine approximating the human decision methods after several consultations with medical specialists.

The second project called MI Elektronikus Almanach was aimed to develop a publically available knowledge base of artificial intelligence topics in Hungarian. My main participation in the project was the development of a competition framework based on Jason, an interpreter of the AgentSpeak language. I created a virtual environment in Java, simulating a world with two agent teams competing. The goal of the agents in the game is to collect the food randomly falling to the ground. This game was successfully used as an optional homework in the education of the subject Artificial Intelligence for the 3rd grader BSc students on information technology. I also contributed the project with creating animated demonstrations for several informed and uninformed search algorithms.

References

[1] P. Eredics. Measurement for Intelligent Control in Greenhouses, 7th International Conference on Measurement, Smolenice Castle, Slovakia, pp 178-181, 2009.

[2] Péter Eredics. Short-Term External Air Temperature Prediction for Intelligent Greenhouse by Mining Climatic Time Series, 6th IEEE International Symposium on Intelligent Signal Processing, Budapest, pp 317-322, 2009.

[3] P. Eredics, T.P. Dobrowiecki. Hybrid Knowledge Modeling for an Intelligent Greenhouse, 8th IEEE International Symposium on Intelligent Systems and Informatics, Subotica, Serbia, pp 459-463, 2010.

[4] P. Eredics, T.P. Dobrowiecki. Neural Models for an Intelligent Greenhouse - The Heating, 11th IEEE International Symposium on Computational Intelligence and Informatics, Budapest, Hungary, pp 63-68, 2010.

[5] P. Eredics, T.P. Dobrowiecki. Data Cleaning for an Intelligent Greenhouse, 6th IEEE International Symposium on Applied Computational Intelligence and Informatics, Timisoara, Romania, pp 293-297, 2011.

[6] P. Eredics, T.P. Dobrowiecki. Data Cleaning and Anomaly Detection for an Intelligent Greenhouse. Submitted to Springer-Verlag's Applied Computational Intelligence in Engineering and Information Technology issue.

[7] P. Eredics, K. Gáti, T. P. Dobrowiecki, G. Horváth. Neural Models for an Intelligent Greenhouse - A CMAC Global Greenhouse Model, 12th IEEE International Symposium on Computational Intelligence and Informatics. Budapest, Hungary, pp 177-182, 2011.

DETECTION OF COMPLEX ACTIVITIES USING AAL ORIENTED SENSOR NETWORK

Péter GYÖRKE
Advisor: Béla PATAKI

I. Introduction

The concept, Ambient Assisted Living (AAL) aims to utilize the recent technological achievements. These made possible to create systems, which can enhance the quality of living, without requiring any direct interaction between the system and the person. Our research aims to create methods and a system, which monitors an elderly person in order to observe his or her daily activities [1][2]. A short-term observation can prevent or detect accidents; a long-term observation can detect the first signs of dementia or other disorders. To specify the exact aims of the activity detector, the following activities are detected in the current version:

- Entering or leaving the apartment
- Sleeping or resting
- Using the toilet or bathroom
- Dining
- Dangerous situations (Not moving for a long time, heater left on, fridge left open)

This paper presents an initial method for detecting activities and based on that experiences we show some future development ideas.

II. Description of the system

Our test environment/laboratory contains motion, contact, load cell (strain gauge) and electric consumption sensors. The contact sensors are used to monitor the state of doors and windows, like the apartment door, or the fridge door. The load sensors are monitoring some furniture, like the kitchen chair and the bed (their outputs are binary). Most of the sensors are "off-the-shelf" products, communicating over a wireless network. The coordinator of the network is a PC, with Linux operational system. The signals of the sensors are transferred to a system bus, called the DBUS, which is designed for inter-process communications. The transfer to the DBUS made by device manager software which is specialized to these sensors. If we would like to introduce a new type (or a brand new) sensor, we have to implement an additional device manager to the new sensors, but modification of the other parts of the system is not necessary. This is possible due to the well-defined signal structure on the bus.

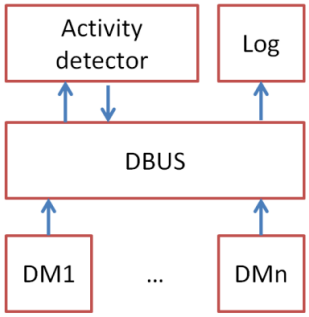


Figure 1: The layered design of the system (DM=Device manager)

The activity detector software is attached to the DBUS and it is subscribed to the relevant sensor interfaces, when a signal on DBUS is sent, a callback function will be activated.

III. Principle of the activity detection

To describe the events we specified an event (e.g. the breakfast) descriptor, looks like:

	Start event	Required sequential event(s)	Required not ordered event(s)	Likely event(s)	Unlikely event(s)	Final event
Event name(s)	*Entering kitchen*	-	*[Using heater, using fridge, using chair]*	-	-	*Leaving kitchen*
Earliest start time	*6:00 AM*	-	-	-	-	-
Latest start time	10:00 AM	-	-	-	-	-
Min. duration [minute]	-	-	[5,1,5]	-	-	-
Max. Duration [minute]	-	-	[20,5,30]	-	-	-

Table 1: The activity descriptor of breakfast

For each activity we have to define the descriptor shown above. For the detection, we used a Finite State Machine (FSM), but it is extended with uncertain states. Uncertain states have a number of sub-states, each sub-state has a probability, these probabilities sum up to 1. When the system enters an uncertain state, one of the sub-states becomes active; the selection of the sub-state is based on its prior probability. Further sensor events can influence the sub-states probability values, therefore they can change the active sub-state. We can influence the probabilities with the elapsed time in the sub-state or with the time of the day as well. The update of the sub-state probabilities uses the following Bayesian method that can be shown most easily by an example. Assume that, when the patient is on the bed, there are 3 possible activities: resting (RT), reading (RD) or sleeping (S) and we have 2 sensors: a motion (M) sensor pointed to the bed and an electric consumption meter, so we can tell when the reading lamp (L) is on. For example the updated probability value of the sleeping (P(S|M,L)) sub-state can determined with the following formula:

$$P(S|M,L) = \frac{P(M,L|S)\cdot P(S)}{P(M,L|S)\cdot P(S)+P(M,L|RD)\cdot P(RD)+P(M,L|RT)\cdot P(RT)} \quad (1)$$

where the P(S) is the prior probability of the sleeping sub-state, and the measurement was moving and lamp on (M,L). In the nominator the prior probability of the sub-state is multiplied with the conditional probability of the measurement. The denominator only contains the normalization, to ensure:

$$P(S) + P(RD) + P(RT) = 1 \quad (2)$$

The conditional probabilities of the measurements can be determined with statistical methods, or if we are lack of samples we fill the conditional probability table (CPT) based on earlier experiences. With this simplification the resulted values are not probabilities, rather scores. For example:

	Moving	~Moving
Sleeping	P(M|S)=0.1	P(~M|S)=0.9
~Sleeping	P(M|~S)=0.7	P(~M|~S)=0.3

Table 2: An example of CPT for binary case

In more complex instances the CPT can be very large; to avoid filling all values we made another simplification in the activity detector software. By the configuration of the software, the expert user

only has to give that the specific sensor event increases or decreases the score of a sub-state. If we have more sensors of the same type, we can assign weight values for the sensors. From these inputs, the CPT can be generated automatically, after that we can still modify particular values, if it is necessary.

As we mentioned earlier the score of a sub-state is influenced by time too. We achieved this by adding time as a virtual sensor, so there is a time related score in the CPT's. This score is a function of the elapsed time in the actual active sub-state and time-of-the-day, this two parts are calculated separately. To get the aggregated time related score, these two values are multiplied. For the calculation of each score, we composed two logistic functions (Fig. 2.), so the time related score is calculated from 4 logistic curves. There are 5 time related parameters for each sub-state:

- Earliest start time (center of the 1st curve)
- Latest start time (center of the 2nd curve)

} Time-of-the-day related score (Fig. 2.)

- Minimal duration (center of the 3rd curve)
- Maximal duration (center of the 4th curve)

} Elapsed time related score

- Steepness of the logistic curve (usually constant for all sub-states)

It is possible to make time independent sub-states probabilities, e.g. bathroom usage can happen any time in a day.

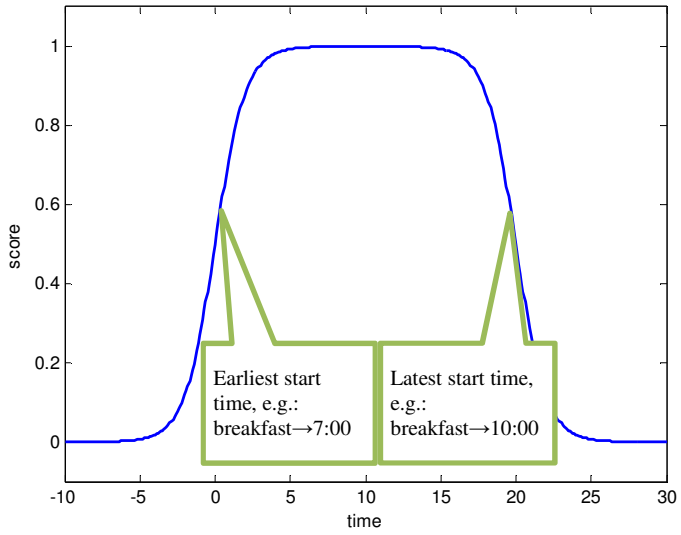


Figure 2: The logistic functions for the calculation of the time-of-the-day related score

Another extension is that we can add conditions for state transitions, a transition can only fire, when a particular state was active. This is useful when there is a precondition for an activity. The FSM can model these preconditions without conditional state transitions, but it needs more states, and the complexity of the FSM should be limited, because a human expert has to handle it.

The output of the activity detector is a DBUS signal for each activity we would like to detect. We can send event signals and activity signals, the difference between them is that the activity signal contains a duration parameter, the event signals are related to momentary events. We can attach an event signal sending method to every state transition, or an activity signal sending method to every state leaving event, where we can send the elapsed time in the state.

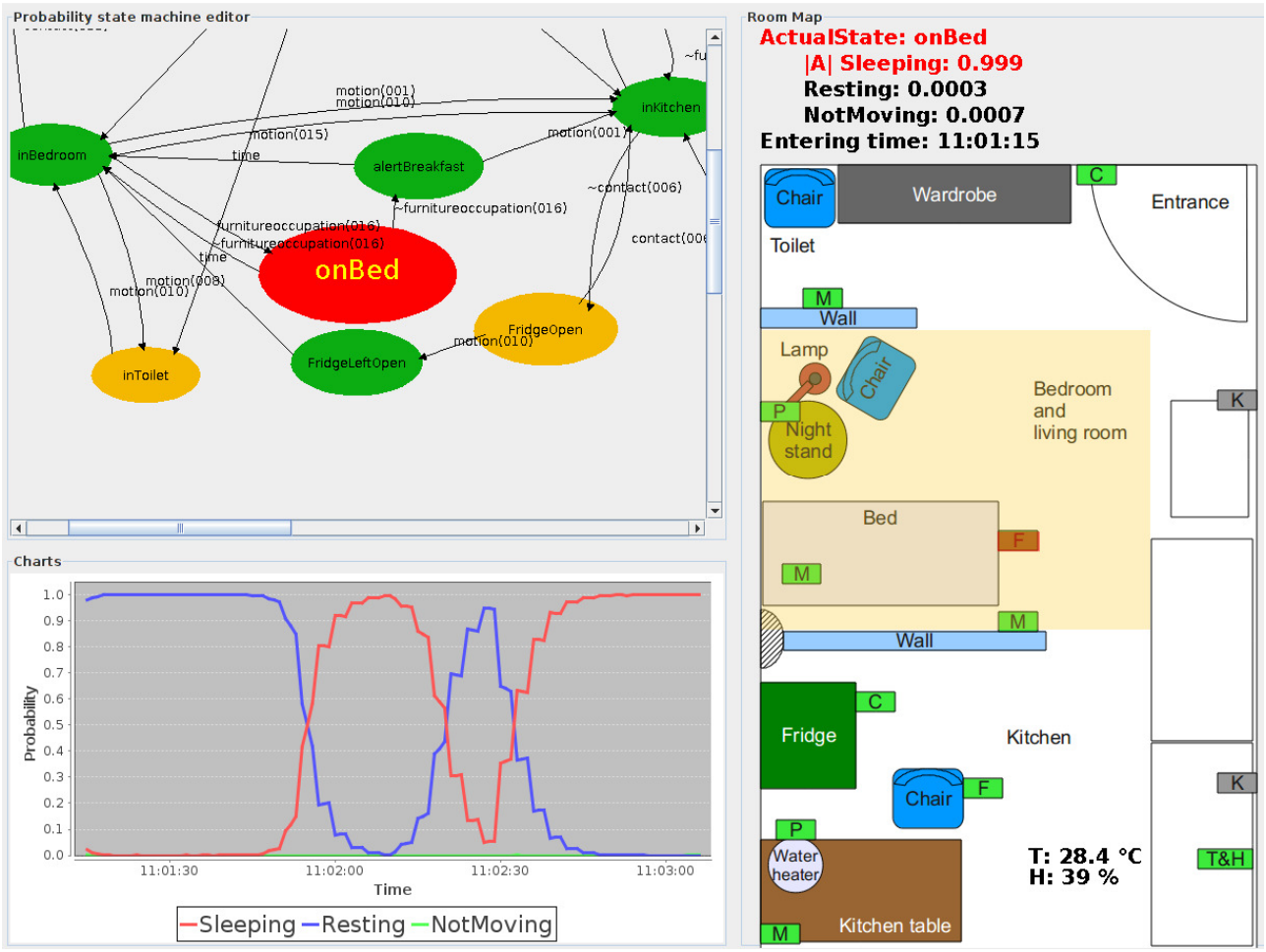


Figure 3: The screenshot of the activity detector

IV. Conclusion and Further Work

This system was successfully tested in our laboratory, and it can detect activities we mentioned above. We used a 10 minutes long test scenario to validate the activity detector's proper operation. A test patient is entered the apartment, then started activities like sleeping, using the bathroom and having breakfast. We implemented three alert events: the system is alerted the patient when the fridge door is left open and an alert was sent to the observers when the patient was in the bed, but no movement happened for a given time. At the end of the test, the patient left the apartment, and an SMS alert was sent to the observers.

The main drawback of the FSM model is that, the number of states can explode, when we would like to detect parallel activities or asynchronous events. A second version of the activity detector is being developed and analyzed. It models the activities in the apartment as probability processes. The outputs of the processes are connected to an evaluation module, which output is the probability of the current activity.

References

[1] Philipose, M., Fishkin, K., Patterson, D., Perkowitz, M., Hahnel, D., Fox, D., and Kautz, H. "Inferring activities from interactions with objects." *IEEE Pervasive Computing Magazine 3*, 4 (Oct.–Dec. 2004),

[2] Emmanuel Munguia Tapia Stephen S. Intille Kent Larson, "Activity Recognition in the Home Using Simple and Ubiquitous Sensors", *Lecture Notes in Computer Science*, 2004, Volume 3001/2004, 158-175

BASICS OF BEST LINEAR APPROXIMATION

Péter Zoltán CSURCSIA
Advisors: István KOLLÁR and Johan SCHOUKENS

I. Introduction

The aim of this paper to present one of the possible ways to handle static, weakly nonlinear time invariant systems using a simple method called best linear approximation (BLA), as an extension of least squares estimation.

II. Weak nonlinearities

To get a basic definition of linearity refer to linear time invariant (LTI) systems. A system is linear if it satisfies the principle of superposition. If the input is denoted by *u* and output is denoted by *y* then superposition means:

$$y = f\big((a+b)u\big) = af(u) + bf(u) = (a+b)f(u) \tag{1}$$

If the system does not satisfy the assumptions above, the behavior of the observed system is nonlinear. Let us define furthermore a weakly nonlinear system as a nonlinear whose nonlinear part can be represented by following equation:

$$y = cu^n, \text{ where n} \leq 3 \text{ and c is a constant} \tag{2}$$

Furthermore, a static nonlinear (STNL) system differs from an LTI system in that an STNL system transfers energy from one (excited) frequency to other (excited or non-excited) frequencies, which is not true in the case of LTI systems.

III. Estimation of weakly nonlinear systems with Best Linear Approximation

The BLA of a nonlinear system minimizes the mean square error between the true output of nonlinear system and the output of the linear model (represented by the impulse response function *g(t)*). For instance this definition is in time domain:

$$\min\{\|\tilde{y}(t) - g(t) * \tilde{u}(t)\|_F^2\} \tag{4}$$

where * is the convolution product [5], and $\tilde{y}(t) = y(t) - \mathbb{E}\{y(t)\}$, $\tilde{u}(t) = u(t) - \mathbb{E}\{u(t)\}$, for further details, see Section III.B.

A Theoretical structure and the basic assumptions

In Fig. 1 the theoretical structure of BLA is considered with the following components [7]:

$\mathbf{G}_{\text{Linear}}$ represents the true underlying linear system if it exists (otherwise it is zero). Its response is independent of the power spectrum of the excitation signal (because of the linearity).

$\mathbf{G}_{\text{Bias}}$ denotes the systematic nonlinear contribution whose spectrum is smooth [3][6][7]. Its value is fixed in the same excitation set.

Thus $\mathbf{G}_{\text{BLA}}$ (as an addition of $\mathbf{G}_{\text{Linear}}$ and $\mathbf{G}_{\text{Bias}}$) has also smooth frequency response function (FRF) [5], which will be the same for the same excitation [3].

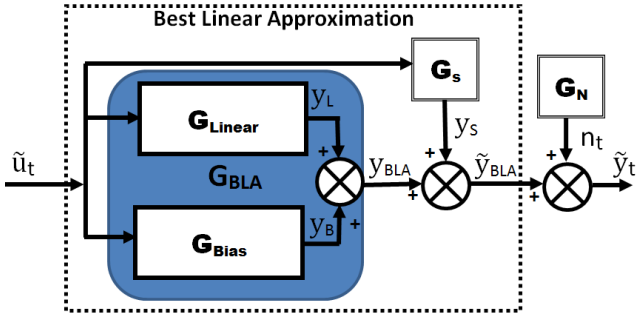


Figure 1: The theoretical structure of the Best Linear Approximation

$\mathbf{G}_S$ is modeled as a 'half-stochastic' nonlinear noise generator. The mathematic definition of $\mathbf{G}_S$ is the difference between the estimated (modeled) and the true underlying system. In this sense half-stochastic means that its behavior is partly fixed and partly random. It is deterministic because for the same excitation set (e.g. for the same multisine excitation [5][7]) its value is fixed. It is stochastic because different parameter values belong to different excitation set. $\mathbf{G}_S$ has the following properties:

- it has circular complex normal distribution with zero mean (stochastic part of $\mathbf{G}_S$)
- same smooth power spectrum for all excitations in the same set (deterministic part of $\mathbf{G}_S$),
- even and odd nonlinearities can be contributed

In other words, $\mathbf{G}_{Bias}$ delegates the nonlinear part what we can model with this paradigm (see also Section IV.) and $\mathbf{G}_S$ represents the (non)linear error.

The properties of the nonlinear parts $\mathbf{G}_{Bias}$ and $\mathbf{G}_S$ are depending on:

- energy level of each excited frequency,
- the probability density function (pdf) of the input excitation, see Fig 2.a.,
- in general: even and odd nonlinear distortions of the observed system, see Section III.C.,
- type of the disturbing noise and its impact point [5]

Noise n_t represents the 'regular' noise (observation error) at the system output. In this article it is assumed that n_t has zero mean normal distribution with variance σ^2, i.e. $n_t \sim \mathcal{N}(0, \sigma^2)$.

Thus, assuming that there is no measurement or round-off error, the steady state frequency transfer function is:

$$\hat{G}_{BLA}(j\omega) = G_{Bias}(j\omega) + G_{Linear}(j\omega) + G_S(j\omega) = G_{BLA}(j\omega) + G_S(j\omega) \qquad (5)$$

With the 'noise transfer function' G_N for the whole observed system the transfer function is:

$$\hat{G}_{obs}(j\omega) = \hat{G}_{BLA}(j\omega) + G_N(j\omega) \qquad (6)$$

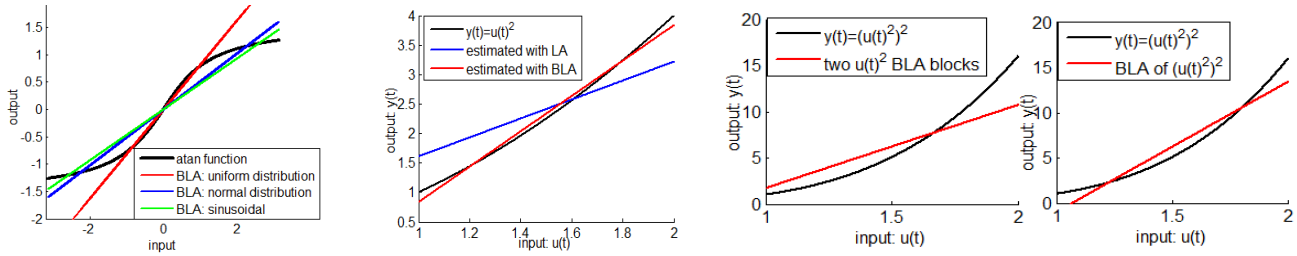
B Importance of removing the mean values and cascading blocks

In case of BLA the DC (or more general the mean) values of the input and output signals must be removed. This is the best linearization (in rms sense) around the operating point, so this is one of the important differences between the regular LS and the BLA. Figure 2.b. shows an example for the simulated, noiseless STNL system of characteristic $y(t)=u(t)^2$ with uniformly distributed input between 1 and 2. With the mean values $\bar{y}, \bar{u}$ the estimated least squares regression line (LSR) and BLA outputs are (consider in Eq. 8. that the original input mean is removed from the input, but the original output mean is added to output value):

$$\hat{y}_{LSR,t} = u_t \hat{g}_{LSR} = u_t \frac{\mathbf{Y}^T\mathbf{U}}{\mathbf{U}^T\mathbf{U}} \qquad (7)$$

$$\tilde{y}_{BLA,t} = \tilde{u}_t \hat{g}_{BLA} = (u_t - \bar{u}) \frac{(\mathbf{Y}-[\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_N]\bar{y})^T(\mathbf{U}-[\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_N]\bar{u})}{(\mathbf{U}-[\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_N]\bar{u})^T(\mathbf{U}-[\mathbf{e}_1,\mathbf{e}_2,\ldots,\mathbf{e}_N]\bar{u})} + \bar{y} \qquad (8)$$

The BLA of the cascade of two nonlinear systems is not equal to the cascade of BLAs of each nonlinear system separately. An example is shown in Fig. 2.c and in Fig. 2.d.



a) b) c) d)

Figure 2: a) Estimation of atan function between $-\pi$ and π with different pdf (uniform, normal, sinusoid) of input signal. b) Estimation of $y(t)=u(t)^2$ with LA and with BLA. c) and d) Estimates of $y=(u(t)^2)^2$ with BLA as cascaded $y=u(t)^2$ two blocks and one $y=(u(t)^2)^2$ block.

IV. Effect of Gaussian excitation and noise

In this Section the effect of Gaussian-like signals is studied. For Gaussian noise (and for every kind of odd random phase multisine [7], see it later on) excitations, the BLA of static nonlinearity is static and independent of the coloring of the power spectrum and the nonlinear G_{Bias} term is independent on even nonlinearities.

However, the BLA of a static nonlinearity is dynamic for non-white, non-normally distributed inputs. In both (Gaussian and non-Gaussian) cases the variance of the BLA measurement depends on the coloring of the power spectrum. Filtered white non-Gaussian noise behaves asymptotically (as the length of the impulse response of the filter tends to infinity) as Gaussian one.

If the length of the random phase multisine is sufficiently large [7], then the systematic nonlinear contribution G_{Bias} does not depend on harmonics of the spectrum, as long as the equivalent power frequency band remains the same. However, the stochastic nonlinear noise part of the BLA G_S strongly depends on the harmonic content in all cases.

V. The Effects of two dimensional averaging

In this article it is assumed that the observed system is static, weakly nonlinear disturbed only at the output with Gaussian noise, and the excitation signals are Gaussian-like [2][7].It is very important that G_{BLA}, G_S and G_N satisfy the assumptions given in the Section III.

In order to estimate the whole observed system $\hat{G}_{obs}(j\omega)$ it will be needed to average over *p* periods (belonging to the same excitation set) and over the *m* different realizations (with different excitation set), please see Fig. 3. For the observed system with the notations above, in frequency domain:

$$\hat{G}_{obs}^{[m][p]}(j\omega) = \frac{\hat{Y}_{obs}^{[m][p]}(j\omega)}{U_{exc}^{[m]}(j\omega)} = G_{BLA}(j\omega) + G_S^{[m]}(j\omega) + G_N^{[m][p]}(j\omega) \qquad (8)$$

First, let us average over the *p* periods in the same excitation set with leaving the transient term. If *p* is sufficiently large then the expected value of the observed system (originated from the law of large numbers [4] and the distribution of n_t):

$$\mathbb{E}\left\{\hat{G}_{obs}^{[m]}(j\omega)\right\} = \frac{1}{p}\sum_{i=1}^{p} G_{obs}^{[m],i}(j\omega) = \frac{1}{p}\sum_{i=1}^{p}\left(G_{BLA}(j\omega) + G_S^{[m]}(j\omega) + G_N^{[m],i}(j\omega)\right) = \qquad (9)$$

$$= G_{BLA}(j\omega) + G_S^{[m]}(j\omega) + 0 = G_{BLA}(j\omega) + G_S^{[m]}(j\omega)$$

and so the variance (for further details see [7]):

$$\mathrm{Var}\left\{\hat{G}_{obs}^{[m]}(j\omega)\right\} = \sum_{i=1}^{p} \frac{\left|G_{obs}^{[m],i}(j\omega) - \hat{G}_{obs}^{[m]}(j\omega)\right|^2}{p(p-1)} \qquad (10)$$

Second, let us average over the *m* different realization. If *m* is sufficiently large, then:

$$\mathbb{E}\{\hat{G}_{obs}(j\omega)\} = \frac{1}{m}\sum_{j=1}^{m}\left(\hat{G}_{BLA}(j\omega) + G_S^j(j\omega)\right) = \hat{G}_{BLA}(j\omega) + 0 = \hat{G}_{BLA}(j\omega) \qquad (11)$$

and thus the variance is:

$$Var\{\hat{G}_{obs}(j\omega)\} = \sum_{j=1}^{m}\frac{\left|\hat{G}_{obs}^{j}(j\omega) - \hat{G}_{obs}(j\omega)\right|^2}{m(m-1)} = Var\{G_N(j\omega)\} + Var\{G_S(j\omega)\} \qquad (12)$$

where the variances of the regular and nonlinear noise are (originated from the distribution properties):

$$Var\{G_N(j\omega)\} = \frac{Var(G_N^{[m][p]}(j\omega))}{pm} \text{ and } Var\{G_S(j\omega)\} = \frac{Var(G_S^{[m]})}{m} \qquad (13)$$

This means that the influence of the noise and non-linear contribution can be decreased with this special two dimensional averaging, as a final result the $\hat{G}_{obs}(j\omega)$ (BLA estimator) will be provided. In [7] it has been shown that the above statements are only true with the condition and assumptions declared in Section III, when the number of periods is greater or equal than two ($p \geq 2$) and the different realizations are more than six ($m > 6$).

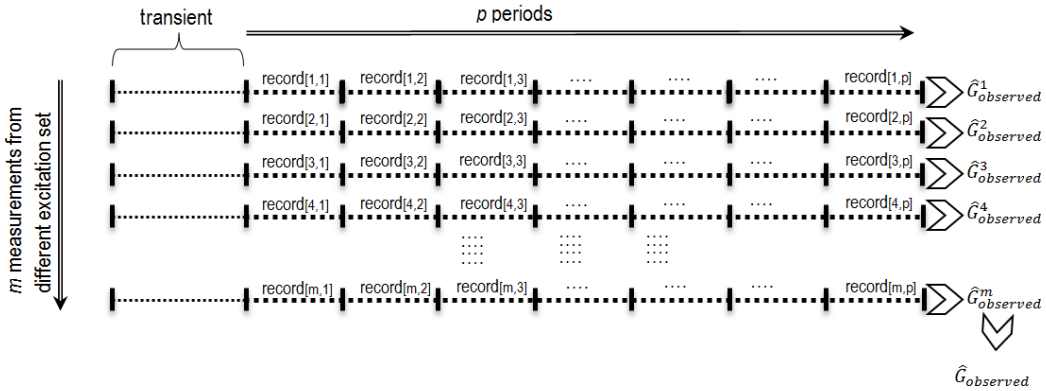


Figure 3: Evaluation of $\hat{G}_{obs}(j\omega)$ with help of two dimensional averaging

VI. Summary

The proposed Best Linear Approximation Estimator is very useful for weakly static nonlinearities because it is originated from linear theorem of plant modeling. It is simple to use without advanced knowledge in statistics and good fitting possible for several kinds of non-linear systems (e.g. Wiener-Hammerstein systems [6]).

However, its usability is guaranteed for this special class of nonlinearities only. Furthermore, when the excitation set is not Gaussian-like, then $\hat{G}_{obs}(j\omega)$ can be more complicated and dynamical.

References

[1] W. V. Moer, Y. Rolain, "Best Linear Approximation: Revisited," I2MTC 2009, Singapure, 5-7 May 2009

[2] J. Schoukens, T. Dobrowiecki, R. Pintelon, "Parametric and non-parametric identification of linear systems in the presence of nonlinear distortions-a frequency domain approach," IEEE Trans. on ACl, vol. 43, pp. 176–190, Feb. 1998

[3] L. Lauwers, "Some practical applications of the best linear approximation in nonlinear block-oriented modeling," ISBN 978-94-9069-56, Uitgeverij University Press, 2011

[4] C. W. Helstrom, "Probability and stochastic processes for engineers," Macmillan Coll Div., 1991

[5] R. Pintelon, J. Schoukens: "System identification, a frequency domain approach," IEEE Press, 2001

[6] L. Lauwers, J. Schoukens, R. Pintelon and M. Enqvist, "Nonlinear structure analysis using the best linear approximation," Proceedings of ISMA2006, pp. 2751-2760, 18-20 September, 2006

[7] J. Schoukens, R. Pintelon, Y. Rolain : "Mastering system identification in 100 exercises," ISBN: 978-0-470-93698-6, Wiley-IEEE Press, 2012

LOCATING CLAVICLES ON CHEST RADIOGRAPHS

Áron HORVÁTH
Advisor: Gábor HORVÁTH

I. Introduction

The analysis of chest radiographs usually starts with the enhancement of image visibility. Bone suppression is an emerging area in this field [1]. Without bone segmentation only poor results can be achieved [2]. In this paper a robust solution is presented for clavicle segmentation.

II. Algorithm description

The segmentation procedure starts with the placement of clavicle templates on the image relative to the lung (see the second image of Figure 2). These shapes are first forced to fit into the range of plausible clavicles and then forced to fit to the image. These steps are then repeated for a fixed number of iterations, as it can be seen in Figure 1. The final result is selected from the outcome of the iterations.

It can be observed on the images of Figure 2 that there are numerous shapes that did not converge to real clavicles. This is due to the high dependency of convergence on the starting position. A unique position which would guarantee the convergence on every image could not be found, therefore it is required to use several shapes simultaneously.

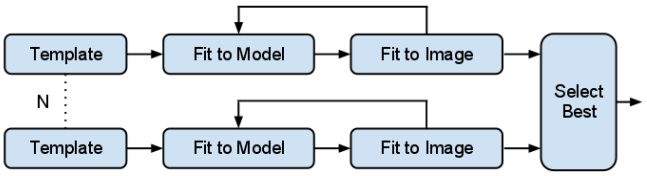


Figure 1: Flowchart of the algorithm

Finding proper initial positions is crucial to reach high accuracy. From a main template, which depends on the size and position of the lung, copies were created by translation and rotation. The goal was to find the translation and rotation parameters which provide the best coverage of the plausible clavicle locations. The parameter space was sampled uniformly, and those points which could be substituted by another point were sorted out. A point can be substituted if there exists another configuration which performs almost as good on each individual image but reaches a higher result on average. This reduction step yielded ten starting positions for a set of 332 clavicles.

To encode the shapes a Point Distribution Model was used. Ten dimensions proved to be enough to model the variations of the 332 training clavicles including translation, rotation and scaling. To regularize a shape, it is enough to fit it to this model, constrain the shape parameters and map back to the space of point enumerations.

Fitting to the image was realized by custom-made active contour algorithm based on dynamic programming. The algorithm produces robust results as it finds the optimal solution of fitting a curve to the image. To improve robustness further the method exploits the fact that the two curves bordering the clavicle are quasi parallel.

Another key point of the algorithm is the selection of the best-fitting shape out of multiple results at the end of the procedure. For this purpose three different approaches were evaluated. The first method selects the curve that fits the best to the image. The second approach assumes that the majority of the curves will converge towards the real clavicle, thus it selects the one that overlaps with the most of

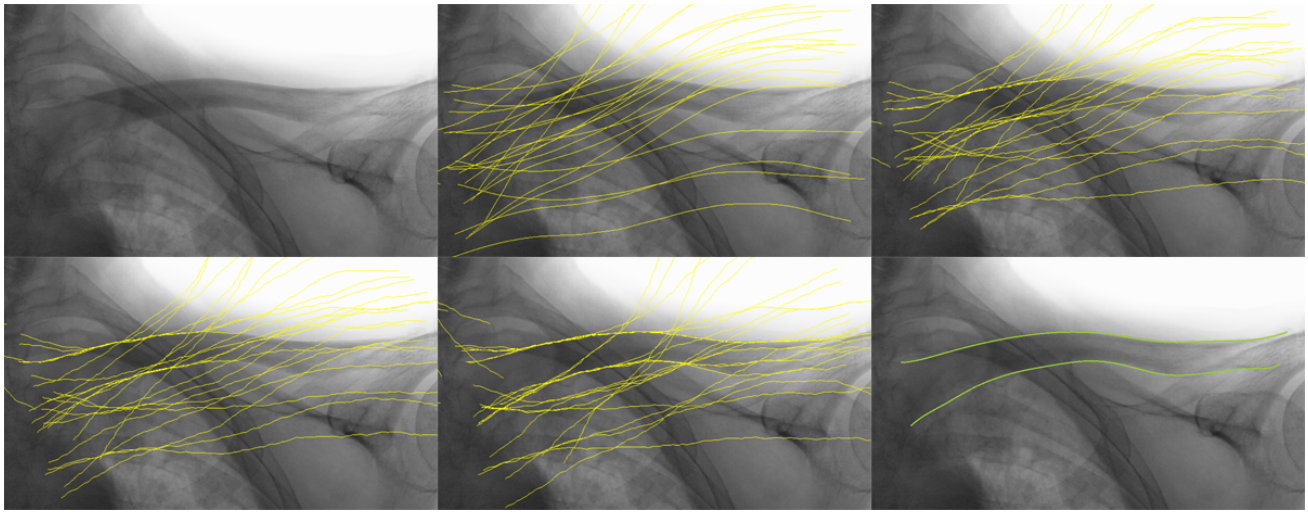


Figure 2: Steps of the algorithm. Top row from left to right: Original image, Templates, After first fitting; Bottom row from left to right: Second fitting, Final fitting, Result

Table 1: Results. Training set / Test set

	best fit	most overlapped	multistage
average error in pixel	5.55 / 5.19	5.91 / 5.76	5.72 / 5.71
$error > 5.8$ (not perfect)	18.67% / 25%	20.78% / 27.5%	19.58% / 27.5%
$error > 10$ (visible error)	6.02% / 5%	5.42% / 10%	6.33% / 5%
$error > 40$ (failed)	1.2% / 0%	1.5% / 0%	1.2% / 0%

other curves, weighted by their individual fitness to the image. The third technique uses the previous measure, but iteratively removes the worst fitting curve and reevaluates the whole set, keeping the last curve as result.

III. Results

The accuracy of the manual delineations can be approximated by the inter-observer agreement. The differences between two set of hand drawn outlines were evaluated over the second batch of images and the average difference was found to be 4 pixels while the maximum difference was 5.8 pixels measured on 2500 x 2500 images. These results imply that the theoretical minimal error of fitness is around 4 pixels, and an error greater than 5.8 pixels means that further improvements are possible.

A comparison of the different selector method types is presented in Table 1. The training set consisted of 166 images, the test set had only 20 images. Due to the small size of the latter the results for the training set are also featured in the table.

The results show that the selector methods that incorporate curve overlapping were not able to outperform the simple fitness-to-image measure. However, the achieved accuracy confirms that this solution can be applied in clinical environment.

References

[1] M. Freedman, S. Lo, J. Seibel, and C. Bromley, "Lung nodules: Improved detection with software that suppresses the rib and clavicle on chest radiographs," *Radiology*, 260(1):265–273, 2011.

[2] A. Khan, T. Onoue, K. Hashiodani, Y. Fukumizu, and H. Yamauchi, "Signal and noise separation in medical diagnostic system based on independent component analysis," in *Circuits and Systems (APCCAS), 2010 IEEE Asia Pacific Conference on*, pp. 812–815. IEEE, 2010.

EFFECTS OF BONE SHADOW REMOVAL ON LESION DETECTION ON CHEST RADIOGRAPHS

Gergely ORBÁN
Advisor: Gábor HORVÁTH

I. Introduction

Lung cancer is one of the most common causes of cancer death. Many cures are only effective in the early and symptomless stage of the disease. Screening can help early diagnosis, but a sensitive – where sensitivity is the fraction of correctly diagnosed positive cases and all positive cases –, cheap and side effect-free method has to be used to enable mass usage. Standard chest radiography meets these requirements, except that current methods have moderate sensitivity. Efficiency can be improved by using a Computer Aided Detection (CADe) system. The most important problem of existing CADe systems is the low positive predictive value. In other words high sensitivity can only be reached at the cost of many false detections. Recently published systems can detect 60-70% of cancerous tumours, while they also mark approximately four false positive regions on each image [1], which allows them to be used only as a second reader.

Usability of CADe systems can be improved either by reducing the number of false detections – to give the examiner less extra work –, or by finding more nodules – to increase sensitivity. The detections of CADe should be also complementary to the findings of radiologists, to better improve sensitivity when radiologists and CADe work in cooperation.

Common causes of detection errors are shadows of the ribcage and the clavicles. On one hand, these structures can conceal the shadows of abnormalities by darkening the image thus reducing contrast. On the other hand, rib crossings on the radiographs sometimes mimic convex structures appearing to be real lesions. Both of these effects seriously affect human examiners and CADe systems too; however, the hiding effect is problematic mostly to radiologists and physicians while the CADe system suffers rather from false positives due to rib crossings. Previous studies showed that the elimination of bones can improve the accuracy of physicians [2]. Our hypothesis was that properly compensating bone shadows can be also used to reduce the number of the CADe system's false findings.

We expected that "cleaning" the image from bone shadows helps our existing CADe scheme to produce less false positives. As the area originally hidden by bone shadows cannot be perfectly restored based on one radiograph, we also created features which help a supervised classifier to eliminate falsely detected structures caused by bones. In the sequel, we describe the existing CADe scheme, the bone shadow removal algorithm and the modifications to improve detection. Afterwards, we demonstrate the results and compare the system to the one without shadow removal.

II. Materials and Methods

For our existing lesion finder solution we used a three-step scheme. The first step segments the viewable area of the lung. The viewable area reduces the possible set of locations for lesions, as we only target the ones partly or completely inside this area.

The second step, called lesion enhancement, highlights suspicious structures like the target lung nodules and infiltrated areas by using image processing algorithms. After normalization and resizing, various 2D filters are used to enhance different types of lesions. Afterwards, the candidate collection converts the enhanced images into separate and segmented nodule candidates.

The last step reduces the number of false positive findings on the enhanced image with the help of a

classifier. It begins with the calculation of features of candidates serving as an input for the classifier. These features mostly describe texture and geometry, and are used by a supervised learning algorithm, namely a support vector machine.

In the updated system, before the second step ribs and clavicles are first segmented based on a previously calculated rough model, afterwards the segmentation is refined to fit edges while satisfying various constraints. Finally the objects are cleared from the image to restore the background. The segmentation steps are not exposed here, a comprehensive description can be found in [3].

The segmentation data is used to remove the bone shadows from the images in order to enhance the structure of the lung. The elimination is based on creating intensity profiles on vertically differentiated images, which are subtracted from the differentiated original image. An integration step returns to the original domain and produces the bone shadow free image.

To better exploit the results of bone segmentation, three new features were created based on the following observations. First, density variations in bone structure can appear as intensity extrema and may produce false positive findings. Second, areas where bones overlap on the 2 dimensional summation images appear as dark, approximately rhomboid shaped structures frequently recognized by the system as lesions. Third, in the cases where bone shadows cause false positive candidates, the border of the candidate follows the edge of the bone in a relatively large portion. These three observations motivated the following three features respectively.

The first feature calculates the area fraction of the nodule overlapping with a bone structure. The feature can be described formally as

$$\mathrm{BoneOverlap}_i = \frac{|\{(x,y)|(x,y) \in C_i \bigcap (\bigcup_j B_j)\}|}{|\{(x,y)|(x,y) \in C_i\}|}, \tag{1}$$

where C_i contains the points of the i^{th} candidate while B_j means the set of points inside the j^{th} bone structure. $|.|$ means the size of the sets. The second feature calculates the overlap of candidates with bone crossings – the areas where two or more bone segments overlap each other. Using the previous notation, the formula of the feature is

$$\mathrm{BoneXOverlap}_i = \frac{|\{(x,y)|(x,y) \in C_i \bigcap (\bigcup_{j,k} (B_j \bigcap B_k))\}|}{|\{(x,y)|(x,y) \in C_i\}|}. \tag{2}$$

The third feature calculates the fraction of the nodule perimeter that runs near to a bone shadow border. It can be calculated as

$$\mathrm{FollowBoneEdge}_i = \frac{|\{(x,y)|(x,y) \in \delta C_i, \min_j d((x,y), \delta B_j) < 1.5\mathrm{mm}\}|}{|\{(x,y)|(x,y) \in \delta C_i\}|}, \tag{3}$$

where δC_i and δB_j are the endpoint set of C_i and B_j, while d is the Euclidean distance.

III. Results

We tested the system on a private chest X-ray database containing images of 285 patients where 134 of the cases contained at least one malignant lung nodule. Nodule diameter ranged from 2 mm to 98 mm, the average was 23 mm. Most of the malignant cases were validated by CT. The images came from a digital X-ray machine working in daily practice at a Hungarian clinic.

10-fold cross validation with 30 repetitions was used for evaluation. Results are demonstrated on free-response receiver operating characteristic (FROC) curves, showing the sensitivity as a function of average number of false positives produced for each image. On the final output, a CADe drawing was considered to be correct if its centroid – center of gravity – was located inside a physician's marker; otherwise it was labeled as a false positive.

We compared the original CADe system – working on unprocessed images – with two new versions utilizing bone information. The first version of the new system consumed bone-shadow-free images on its input, but involved no other modifications compared to the original system. The second version used –besides the cleared radiographs – the three new features mentioned above. The results can be seen in Figure 1.

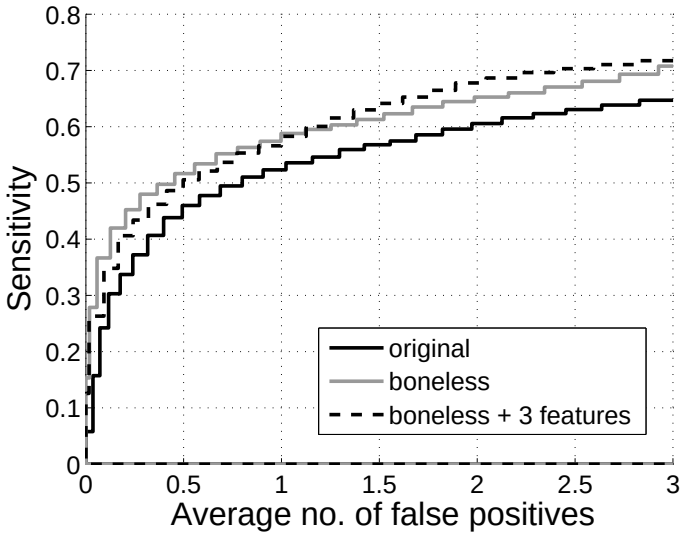


Figure 1: FROC curves of the original nodule detector system (black line), the same system using preprocessed images without bone shadows (gray line) and the system utilizing both the new features and preprocessed images (dashed line). The usage of cleared images greatly improves detection accuracy, while the new features show a slight benefit at important working points.

The improvement when using rib-shadow-free images is clear. At constant 65% sensitivity the number of false positives could be reduced from 2.83 to 1.9. Utilizing the new features, the number of false positives falls further to 1.6. This means that 43% of false detections could be eliminated which is a great benefit for radiologists and physicians having to examine much less false CADe results at screening. Examining the working points at lower sensitivities, the benefit of bone-shadow-free images are still clear; however, the new features cannot improve the accuracy further. Moreover using the new features has a negative effect under 57%, where the output depends stronger on the classifier performance. This can be an effect of more features being used that can reduce the generalization capability of the classifier. However, in everyday practice the working points with higher sensitivities considered to be more useful.

After seeing the benefits in detection accuracy we wanted to see how bone shadows and their removal affect the output distribution of the CADe system. We assumed that bone shadows and especially bone crossings cause many false positives in the original system and less or not at all in the new version. We conducted two tests to support our hypothesis.

In the first test we calculated the fraction of the total area of false findings that overlaps with bones. If we assume no connection between bone shadows and false positive location the fraction should be approximately the same as the fraction of bone shadow area inside the lung area as findings are also restricted to this area. As it can be seen in Table 1 false positives tend to overlap more with bone shadows using the original system while overlap even less than expected on rib shadow free images. This means that the new system is less likely to produce false positives on bone shadows than on other areas. We also checked how often physician markers overlap with bone shadows. The corresponding row of the table shows that physicians tend to avoid bone shadows when marking nodules, just like the new CADe system. We conducted the same measurements for rib crossings and came to similar

conclusions. While the original system puts more false findings on rib crossings, the new system and physicians tend to put less.

Table 1: Area fraction of false positive candidates overlapping with bone shadows.

	Original image	Rib-shadow-free image
Lung area overl. bones	0.489	0.489
FP area overl. bones	0.527	0.446
Phy. marker area overl. bones	0.471	0.471
Lung area overl. bone Xs	0.114	0.114
FP area overl. bone Xs	0.166	0.103
Phy. marker area overl. bone Xs	0.079	0.079

In the second test we analyzed the independence of a candidate being false positive and its overlap with bone shadows as two random variables. The first one is already binary while the overlap feature was quantized to three bins to discretize it. We used Pearson's chi-squared test to check independence and demonstrated the resulting p-values in Table 2. We ran the test for the original system, the system using rib-shadow-free images and the system also using the new features. Using a predefined significance level of 1% the first two cases are clearly significant meaning that false positiveness depends on – or at least correlates with – the overlap with bones while in the third case we can no longer be sure that the relation exists. However, we should note that a p-value of 0.049 suggests further investigation. We ran the same test using the bone crossing overlap as the second variable. Here, using the original system shows a clear correlation, while the results for the versions exploiting bone information are insignificant. These results do not contradict with our original hypothesis; however, the proof of independence needs other studies.

Table 2: Resulting p-values of independence tests between bone overlapping and false positiveness.

	Original image	Rib-shadow-free image	Rib-free + 3 features
Overlap w. bones	0.0001	0.0002	0.049
Overlap w. bone crossings	0.0015	0.61	0.2

IV. Conclusion

Our measurements proved that the removal of bone shadows can improve lesion detection accuracy on chest radiographs. The absence of these dark shadows reduces the number of misleading structures on the image while some newly created features help the classifier to distinguish between remainders of bones and real lesions. We also showed that the output of the newly created CADe system seems to have less dependency on bone location compared to our previous solution.

References

[1] S. Chen, K. Suzuki, and H. MacMahon, "Development and evaluation of a computer-aided diagnostic scheme for lung nodule detection in chest radiographs by means of two-stage nodule enhancement with support vector classification," *Medical Physics*, 38:1844, 2011.

[2] F. Li, T. Hara, J. Shiraishi, R. Engelmann, H. MacMahon, and K. Doi, "Improved detection of subtle lung nodules by use of chest radiographs with bone suppression imaging: Receiver operating characteristic analysis with and without localization," *American Journal of Roentgenology*, 196(5):W535–W541, 2011.

[3] S. Juhász, Á. Horváth, L. Nikházy, G. Horváth, and Á. Horváth, "Segmentation of Anatomical Structures on Chest Radiographs," in *XII Mediterranean Conference on Medical and Biological Engineering and Computing 2010*, R. Magjarevic, P. D. Bamidis, and N. Pallikarakis, Eds., vol. 29 of *IFMBE Proceedings*, pp. 359–362. Springer Berlin Heidelberg, 2010, 10.1007/978-3-642-13039-7_90.

OPTIMAL CONTROL WITH REINFORCEMENT LEARNING USING GAUSSIAN MIXTURE MODELS

István ENGEDY
Advisor: Gábor HORVÁTH

Introduction[1]

Reinforcement learning (RL) is often used to solve optimal control problems. In those problems the goal is to calculate a control policy that is optimal with respect to a criterion function. Such problems are the well-known mountain car problem, or the inverted pendulum problems, which are benchmark problems for optimal control algorithms. The effectiveness of reinforcement learning is most apparent in complex nonlinear or even discontinuous control problems, non-stationary control problems, where classical control methods cannot be used.

In this paper we present a reinforcement learning based method that is able to solve optimal control problems in continuous state spaces, which are difficult to be managed by classical tabular reinforcement learning methods. Usually, function approximators, like neural nets or linear function approximators are used to overcome this problem. They are practical, not just because they can store a continuous value function, but because they can also interpolate the value function between visited states, so the agent has some knowledge about states it never visited.

However there is a possible problem with the function approximation approach. Some of these function approximators are based on iterative algorithms themselves, and they also need all the training points during the training, which are usually not available during RL. Additionally the two iterative learning algorithms (the RL algorithm and the function approximator) might cause convergence problems as they are working together simultaneously [2].

We are using an incremental probabilistic function approximator to store the value function, that doesn't need all the training points during training. This function approximator method is a novel approach in the field of machine learning, and it has been used successfully with reinforcement learning [3]. In the followings we will present a solution to solve MDP's in continuous state spaces that utilizes Q-learning with a function approximator called Incremental Gaussian Mixture Network (IGMN).

The rest of this paper is organized as follows. Section II describes each method in detail. In section III the unified method is presented. In section IV preliminary test results are shown on the inverted pendulum problem.

I. Methods in detail

A Reinforcement Learning

In classical reinforcement learning the problem is defined as a Markov Decision Process (MDP): the agent has to choose between the possible actions at a given state, and based on the chosen action, it perceives the next state and a reward signal from the environment. The aim of the agent is to find the optimal policy π^*, which is when followed, maximizes the expected reward.

Most reinforcement learning algorithms use a value function or Q-function, which gives the expected discounted total reward at a given state for a given action, following policy π. The policy is determined based on this Q-function: in every state the action with the greatest Q-value is to be selected. It is proven that the optimal policy is equal to the greedy policy – always selecting the best

[1] This summary is based on *Optimal Control with Reinforcement Learning using Reservoir Computing and Gaussian Mixture*, submitted to I2MTC'12, Graz, Austria

action – over the Q-values of the optimal policy [1]. Because of this, most reinforcement learning algorithms focus on determining the Q-values of the optimal policy, by means of learning.

The Q-function is estimated during the learning, by making a series of actions and perceiving the reward meanwhile, using a reinforcement learning update rule, like Q-learning (1) or SARSA. Both of these rules are based on the temporal difference learning, the Bellman equation.

$$\Delta Q(s_t, a_t) = \alpha[R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (1)$$

In this equation *Q(s, a)* is the Q-value of action *a* in state *s*. R_{t+1} is the perceived reward at the next state. The parameter α is the learning rate, γ is the discount factor, a real number in *[0, 1]*. That is needed to avoid divergence of the total reward in case of non-episodic problems. This learning rule backs up the perceived reward to the Q-values of the previous states.

$$\pi(s) = \begin{cases} \text{argmax}_a \, Q(s, a) \,, P(\varepsilon) \\ \text{random action}, \; P(1 - \varepsilon) \end{cases} \quad (2)$$

In order to actually improve the value function, sometimes new random moves must be made, to explore new actions and new states. This is called the exploration versus exploitation problem. Usually it is solved with stochastic policies, which tends to choose the best action, but not deterministically. A very simple, yet effective such policy is the ε-greedy policy (2), which takes the best action with ε probability, and a random action with $(1 - \varepsilon)$ probability.

B Incremental Gaussian Mixture Network

Incremental Gaussian Mixture Network (IGMN) [4] and its predecessors, Incremental Gaussian Mixture Model (IGMM) and Incremental Probabilistic Neural Network (IPNN) [5] are machine learning methods, using a mixture of multivariate normal probabilistic distribution functions. All of these methods are incremental, in the sense that they can learn by presenting data points only once to these algorithms, without the need to store the data points. Both IPNN and IGMN are function approximators, while IGMM is a trainable model, using unsupervised learning. While IPNN considers the probability distribution functions as a neural network, IGMN rather use probabilistic inference to calculate the output, thus it is more like a Bayes-network. In the followings, we will use the IGMN method only. Both IGMN and IPNN were successfully used in reinforcement learning problems as function approximators for the Q-value [4], [5].

The Gaussian mixture model inside IGMN is over the joint space of the input and output of the network. The output is calculated from the input using Bayesian inference. This model also makes it possible to infer any part of the input or the output data from the known data. As the model is trained, new components can be added and irrelevant or spurious components with low prior probability could be removed from the model. The training of the model is done by maximizing the likelihood of each incoming data point by adjusting the parameters in the model.

Each component in the model has a couple of parameters. C_j is the covariance matrix and μ_j is the mean of the j^{th} normal distribution function. *P(j)* is the prior probability, *v(j)* is the age, and sp_j is the accumulator of the j^{th} component. The updating of the model is done as follows. First, the likelihood of each component is calculated from the multivariate normal probability distribution function (3):

$$P(x|j) = \frac{1}{(2\pi)^{D/2}\sqrt{|C_j|}} e^{-\frac{(x-\mu_j)^T C_j^{-1}(x-\mu_j)}{2}} \quad (3)$$

After that, the posterior is calculated based on this likelihood, and the prior (4):

$$P(j|x) = \frac{P(x|j)P(j)}{\sum_{i=1}^{M} P(x|i)P(i)} \quad (4)$$

The rest of the training algorithm is done as follows, according to [5]:

$$v_j(t) = v_j(t-1) + 1 \tag{5}$$

$$sp_j(t) = sp_j(t-1) + P(j|x) \tag{6}$$

$$e_j = x - \mu_j \tag{7}$$

$$\omega_j = \frac{P(j|x)}{sp_j} \tag{8}$$

$$\Delta\mu_j = \omega_j e_j \tag{9}$$

$$\mu_j(t) = \mu_j(t-1) + \Delta\mu_j \tag{10}$$

$$C_j(t) = C_j(t-1) + \Delta\mu_j \Delta\mu_j^T + \omega_j\left[e_j e_j^T - C_j(t-1)\right] \tag{11}$$

$$P(j) = \frac{sp_j}{\sum sp_i} \tag{12}$$

The training is started with an empty model. A new component is added to the model when the following criterion (13) is met:

$$P(x|j) < \frac{\tau_{nov}}{(2\pi)^{D/2}\sqrt{|C_j|}} \; \forall j \tag{13}$$

Here τ_{nov} is a novelty parameter, which tells how much part of the space is actually represented by that distribution, in the sense that a new data point in the outside area bears new information, thus a new component must be created for it. The only parameter at new component creation is the initial covariance matrix, which could reflect the sizes of the space along each dimension.

Components should also be removed from the model after a certain age, if their prior probability is decreasing over time. That means the specific distribution is never selected as a relevant distribution for any data points.

The recall phase of the IGMN in the function approximation case is done using probabilistic inference. Let the data x be the joint of input vector a and output vector b, $x=[a, b]$. The output is calculated as follows (14):

$$\hat{b} = \sum_{i=1}^{M} P(j|a)\left[\mu_{j,b} + C_{j,ba}C_{j,aa}^{-1}\left(a - \mu_{j,a}\right)\right] \tag{14}$$

Where $C_{j,ba}$ and $C_{j,aa}$ are submatrices of the covariance matrix.

II. The proposed system

As we discussed it in the introduction, our aim is to make an intelligent controller for higher level control tasks, specifically the inverted pendulum problem using machine learning algorithms. The core framework of our proposed system is reinforcement learning, with the Q(λ) learning rule. We assume that a simulator is available for the system, which can be used to generate sample runs, and actually the RL can be trained on it.

We use an IGMN to store the Q-values of the actions for each state. So the samples for the IGMN are the joint vectors of the state and action vectors and the actual Q-value. This way we take the assumption, that both the state and action spaces are continuous. The reason to take the joint distribution of the input of the Q-function and the value of it instead of taking it as a function is that

the RL algorithm at the beginning of the training tends to provide very different Q-values for the same state and action based on the trajectory it later follows. It would be impractical to take all of these values as points of the Q-function. These points should be averaged, and that is what the probabilistic inference does, when the points are represented as a joint distribution. Later during the training, sample points in the distribution with spurious Q-values are never selected again, thus their prior probability will decrease, and eventually they will be removed from the model.

The training goes as follows: first the Q-values of all actions in the actual state are calculated based on the IGMN model, using (14). Then based on ε-greedy policy (2), the best action or a random action is selected. After that the system is simulated with the simulator, using the selected action. In the new state, the reward signal is perceived, and the Q-values of the previous state and action are updated, based on (1). To actually update the Q-values, the training method of the IGMN is used. The posterior of each Gaussian are calculated using (3) and (4). Then they are updated according to their relevance to the previous state and action and the newly calculated Q-value with equations (5)-(12). If (13) is met, a new Gaussian is added to the model.

III. Test results

We have tested the proposed system on a limited torque inverted pendulum task, where only the angle variable was observable, the angular velocity was hidden. There were only two actions, to apply the same amount of torque to the pendulum to left or to right. The parameters of the task were selected such way, that the pendulum must make multiple left-right transitions in order to get to the instable equilibrium position.

The reward function was the cosine of the angle of the pendulum. The system was trained over 300 episodes, each episode 200 time steps. In each episode the pendulum was started from a random state.

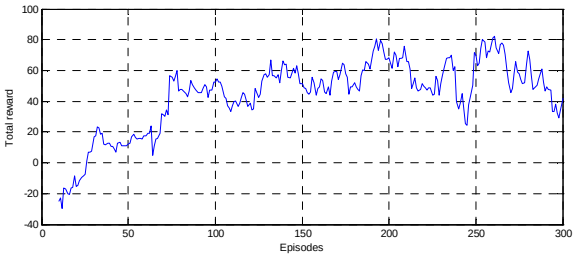


Figure 1. Total reward per episode during the training

As the tests shown, the proposed method was able to improve the performance of the policy, eventually after about 70 episodes of training the inverted pendulum spent most of the time near the goal position. The IGMN part learned 195 Gaussian distributions.

References

[1] Sutton, R., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA (1998)

[2] S. Thrun and A. Schwartz: Issues in using approximation for reinforcement learning. In Proceedings of the Fourth Connectionist Models Summer School, Hillsdale, NJ, 1993. Lawrence Erlbaum Publisher.

[3] A. Agostini and E. Celaya. Reinforcement learning with a Gaussian mixture model. In Proc. Int. Joint Conf. on Neural Networks (IJCNN'10). (Barcelona, Spain), pages 3485–3492, 2010.

[4] Heinen, M.R. "A Connectionist Approach for Incremental Function Approximation and On-line Tasks". Porto Alegre, RS. Ph.D. Thesis. Universida-de Federal do Rio Grande do Sul – UFRGS, 172 p. (2011)

[5] Heinen, M.R., Engel, P.M. "An incremental probabilistic neural network for regression and reinforcement learning tasks". In: K. DIA-MANTARAS; W. DUCH; L.S. ILIADIS, Artificial Neural Networks – ICANN 2010. Berlin, Springer-Verlag, p. 170-179.

DATA ANALYSIS FOR TIME SERIES FORECASTING

Kristóf GÁTI
Advisor: Gábor HORVÁTH

I. Introduction

This paper presents an approach for time series prediction. For the solution of these problems, in general, nothing more is assumed just the time series itself, and by using a nonlinear training algorithm just a black-box model of the system could be trained. Decomposition of the problem may lead to a better understandable model. The decomposition should create more time series, where one of them is the trend and if the signal has periodicity than this decomposition can be done if frequency domain. With these decomposition the other signals will be more and more periodic, however most of the noise will be present in these high-frequency signal. This decomposition can be made in the frequency domain, as selecting the trend, and the signals with different periodicity is relatively easy. With this decomposition the problem is transformed and now the response of the nonlinear training algoritms can be interpreted, so from a black-box model, we achived a solution which is closer to the grey-box model.

This method is applied for a real time series prediction problem. The data contains the shipments of a company. Since this company is present on the stock markets, at the end of every quarter in a year they have to fulfill every order received in the given quarter. Because of the operations at the company, high discounts on the price, there is a spike at the end of a quarter, which means this period of the year are the most challenging to execute without delay in the shipments. This requires precaution at the planning. The problem of this planning is addressed by this paper. Our goal is to predict as soon and as precisely as possible the amount of shipments required to fulfill in a quarter.

There are different products, decomposed into releases and type (base and extension). All of these are the subject of the forecast, however in this paper only their aggregated value will be predicted. This time series may give further help in the prediction process, since it may serve as input for the prediction of its components, and it is more easier to predict, since the aggregated value is assumed to have less noise then its components. An example for this is the life-cycle of a product. The different products are introduced in different quarters, they are eliminated in different quarters, so the ratio of shipment between the different products has a high diversity in time. This effect is averaged out due to the aggregation, as the full amount of shipment remains quite the same. So this aggregated time series may be assumed independent from the different products, and the seasonal trend may be analyzed and forecasted.

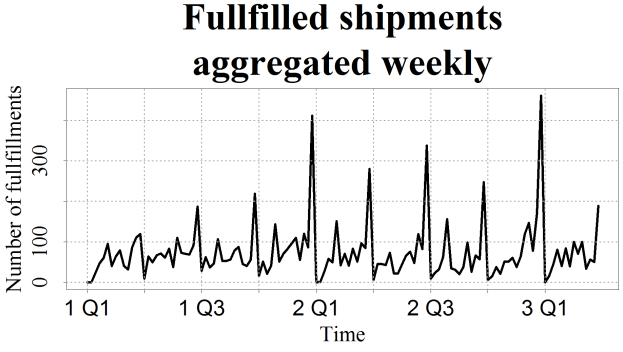


Figure 1: The raw data

The raw data can be seen on Fig. 1. The vertical dotted lines and the corresponding labels identifies

the beginning of a quarter. On the last week of the quarters the spikes are conspicuous. It should be noted that Q4 (4th quarter) has always a higher spike then other quarters.

The code was written almost entirely in R [1], and some code was written in Matlab for convenience. The implementation of SVM was used from the R package `kernlab`. For the identification of the model order the Lipschitz-index [2] was calculated with the `NNSYSID` toolbox.

The paper is organized as follow Section II. describes the preparations made to the data, for the ease of prediction. Section III. show an early version, where the time series was decomposed into parts and Section IV. introduces a different approach for the prediction. Finally Section V. summarizes the results, and shows the remaining open questions.

II. Preparing data

The available data included 9 quarters of shipments. This period was plotted on Fig 1. At the beginning the data was noisy. It contained bad values and NAs. Bad values means the columns slipped so the values of the i-th column were in the $i+1$-st column after some row, or the name of the customer was wrong. The records of the data was in different Excel files, so they had to be merged. Because these was not created at the same time and the products changed during the data gathering period the columns and/or the name of the columns changed in the different files. For example every file used a different format for timestamp (some example: 2009-01-10, 2009.01.10, 10/01/2009, 01102009, 20090110, 2009 1 10, etc.). Some of the records contained missing values. When it was possible these missing values had to be replaced, and most of these could be done by using some simple rules.

The most challenging problem was the time. Since we had to predict for a quarter on a weekly basis, we examined the number of weeks in the quarters and it was 13 or 14. This difference corrupted the periodicity in the data. The following table explains the problem in detail. Here week does not mean full weeks, so if January 1 falls on Sunday it is the first week of the quarter. Problem occurs in those cases when only there are only 13 weeks. This can be handled by adding a ”dummy” week preceding to the quarters first week.

Table 1: Days and weeks of the quarters in a year

Month	**Days in month**	**Days in quarters**	**Weeks in quarters**
January	31	90(91)	13 weeks if January 1 is on Monday or Tuesday and it is not leap year, else there are 14 weeks
February	28(29)		
March	31		
April	30	91	13 weeks if March 1 is on Monday, or it is 14 weeks
May	31		
June	30		
July	31	91	13 weeks if July 1 is on Monday, or it is 14 weeks
August	31		
September	30		
October	31	92	14 weeks
November	30		
December	31		

Before the predictor can be trained, the model order has to be selected. For this purpose, the Lipschitz-index has been calculated. The results are on Fig. 2. The model order was in the range of $[2, 30]$. Based on this plot two comments can be made. First, the periodicity of 14 week is easy to recognize. Second, there are two possible choice for model order based on the plot, at 14 and at 56, the first one means, the predictor uses only the last quarter of the time series, and the latter one means, the predictor will use the data from the last 1 year. Both approach has its advanteges. The first one requires shorter historical data for the generation of the training samples, however the same quarters of the year are much more similar, then the adjacent quarters, so the latter one should indicate a better generalization of the data, and a better prediction.

Further model selection procedures was not applied, the time-lags used for the prediction were selected with trial-and-error method.

III. Decomposition

The first experiments for the prediction was to decompose the time series first. For this, filtering in the frequency domain has been applied. First the spectrum of the time series was calculated using Fourier-transform. The result can be seen, on Fig. 3(a), using a logarithmic scale for the Y-axis. The partition of the spectrum is made so that every spike in the spectrum will belong to a different part. These filtered signals are transformed back to time domain, and the prediction is made on these signals. The assumption is that these filtered signals are easier to predict. The low-pass filtered signal is on Fig. 3(b). These signals are so simple they can be trained to neaural networks. In this case every filtered signal was trained to a different MLP. The original time series can be reproduced with the sum of these signals. The MLPs could learn every signal without error.

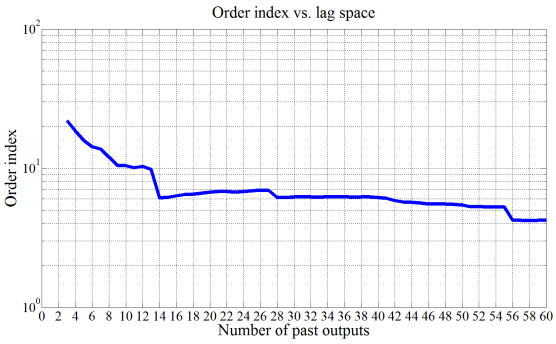
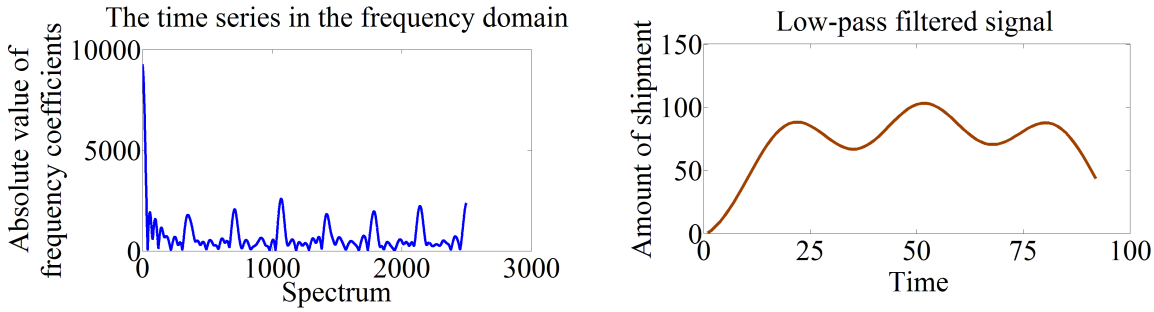


Figure 2: The value of the Lipschitz-index

The training samples could be learned perfectly by the neural networks, however there was a mistake introduced in our model. For the generation of the filtered signals we used the test samples too, so the test results could not be considered as a real measure of performance. The long-term prediction, when the output of the MLPs used as input for the next time step, resulted in a very bad performance. This is because of the Fourier-transform, when a new sample is attained the filtered signals has to be recalculated and the spectrum changes, so the trained signals wil change as well.



(a) The absolute value of the frequency components (b) Low-pass filtered version of the time series

Figure 3: Decomposition of the time series

IV. Aggregation

Because of the bad performance of the previous approach a new method had to be applied. Again, the main problem was the noise present in the time series, see Fig. 1. The possible solution was again the aggregation. So the time series was tranformed, where for every week the corresponding value was the total amount of shipments in the given quarter until the current week. The time series resulted after the tranformation is on Fig. 4

This time series was predicted with an SVM [3]. The hyperparameters (generalization C and tolerance ϵ parameters) were selected again with trial-and-error method. The model order as it was stated

in Section II., two choice of model order is appropriate, 14 and 56. Both are used for prediction in the figure below, Fig. 5. In the first case $[1..5, 10..14]$ time-lags and in the latter case $[1..20, 40..56]$ time-lags were used.

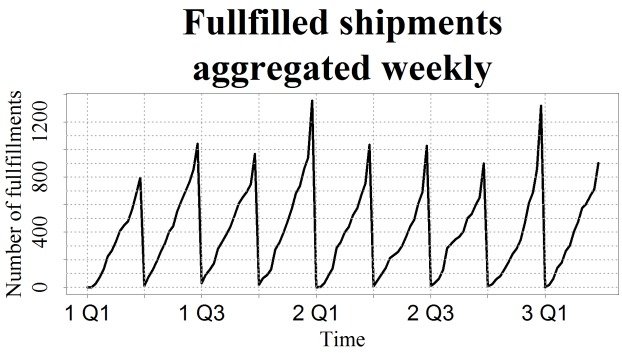
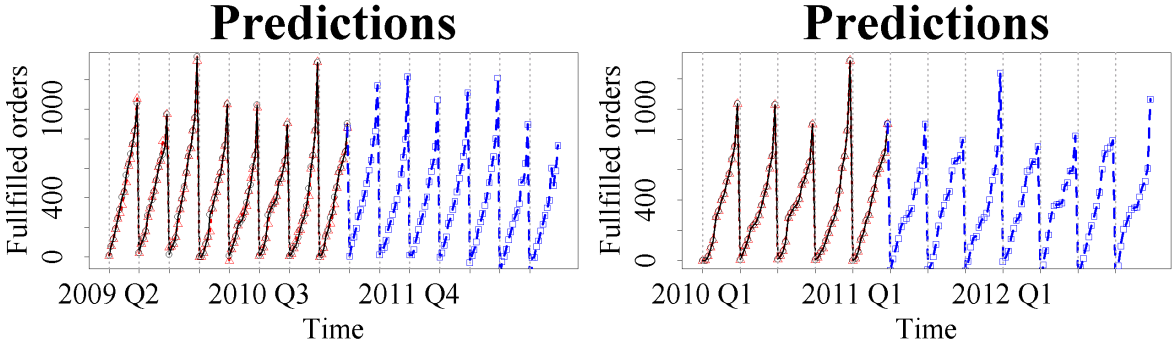


Figure 4: Aggregated time series

As it can be seen, in the first case when the model order was 14, the quarterly trend of the time series could be learned by the SVM, however when the order was 56, not just the quarterly but the yearly trend could be learned, as well. This suggests to use the higher order model, because of the better performance. There was not remarkable difference in time between the two version. The main problem of the higher order model shows up at lower level prediction, where the forecast of the different types is the goal. This is because the main trend at these time series changes faster, and for the first year, we do not possess the necessary amount of data to even train, unlike in the case of the smaller order.



(a) Model order is 14

(b) Model order is 56

Figure 5: Performance of prediction. Black is the training data, red is response of the predictor for the training samples and blue is the actual prediction

V. Conclusion and further work

The main conclusion of this paper is decomposition of time series is generally not a good choice, the information hidden in the data may be lost by the decomposition. Another, not too novel conclusion is, that the selection of regressor has a notable effect on the performance of the predictor.

Beside the achieved results several open question remains. Some of them is to improve the performance of the predictor, by selecting other historical data as inputs for the predictor. Create forecast for ”lower level” prediction, i.e. predict less aggregated time series.

References

[1] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2010, ISBN 3-900051-07-0.

[2] X. He and H. Asada, “A new method for identifying orders of input-output models for nonlinear dynamic systems,” in *Proceedings of the American Control Conference*, pp. 2520–2523, San francisco, CA, USA, June 2–4 1993.

[3] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the fifth annual workshop on Computational learning theory*, COLT ’92, pp. 144–152, New York, NY, USA, 1992. ACM.

THE RELEVANCE VECTOR MACHINE AND AN IN SILICO STUDY OF THE OXYTOCINE RECEPTOR GENE (SUMMARY OF PHD WORK IN 2011)

Peter MARX
Advisor: Peter ANTAL

I. Introduction

Understanding connections in huge amount of data is one of the greatest challenges in bioinformatics. People search databases, and try to find connections between variables. In this short report, I will describe my research in two parts. In the first part, I will describe my work related to Bayesian feature selection in case of continuous target variables and the relevance vector machine (RVM)[1]. In the second part, I will briefly introduce my oxytocine receptor gene related work and my future work.

II. The Relevance Vector Machine

Last year I started to look for existing methods for Bayesian variable selection in continuous cases. The first method I have analyzed was stochastic search variable selection (SSVS)[2]. After I studied the pros and cons of SSVS (it has its limitations in nonlinear cases) I tried to find a method which also works in nonlinear cases. RVM was developed by Tipping. He wanted to extend the famous support vector machine with the following properties:

- get probabilistic solution;
- use non-Mercel kernels;
- get sparser result in case of bigger training sets.

The probabilistic solution is just in that case important, if you want to know the goodness of the result. For example, if you use SVM for classification you get the class of an input element. If you use RVM you get a probability for all classes. RVM gives sparser result but with the newest modification of SVMs we have the possibility to get comparable result. The upgraded SVM method has a little higher error than the first SVM but gives sparser result so we can also find an optimal ratio between sparsity and error using SVMs.

RVM takes the SVM model:

$$y(\mathbf{x};\mathbf{w}) = \sum_{i=1}^{N} w_i K(\mathbf{x},\mathbf{x_i}) + w_0 \tag{1}$$

which we can write in a probabilistic model with added noise $t_n = y(x_n,\mathbf{w}) + \varepsilon_n$:

$$p(t|\mathbf{x},\mathbf{w},\beta) = \mathcal{N}(t|y(\mathbf{x}),\sigma^2) \tag{2}$$

where the standard deviation of the noise is σ^2. To avoid over-fitting we introduce some hyperparameters, to penalize the model complexity. $\boldsymbol{\alpha}$ gives to every weight a hyperparameter and β to variance.

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=1}^{M} \mathcal{N}(w_i|0,\alpha_i^{-1}) \tag{3}$$

$$p(\boldsymbol{\alpha}) = \prod_{i=0}^{N} Gamma(\alpha_i|a,b) \tag{4}$$

$$p(\beta) = Gamma(\beta|c, d) \tag{5}$$

where $\beta \equiv \sigma^{-2}$. In the above equations the value of a, b, c, d can be chosen by the user. You can find detailed information in Tipping (2001).

Learning the hyperparameters goes iteratively and learning the model goes by Bayesian inference.

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)}{p(\mathbf{t})} \tag{6}$$

where

$$p(t) = \int p(t|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2)p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|t)d\mathbf{w}d\boldsymbol{\alpha}d\sigma^2 \tag{7}$$

In equation (6) the normalizing integral cannot be computed analytically so we have to decompose the posterior. The posterior distribution over the weights is the following:

$$p(\mathbf{w}|\mathbf{t}, \boldsymbol{\alpha}, \sigma^2) = \frac{p(\mathbf{t}|\mathbf{w}, \sigma^2)p(\mathbf{w}|\boldsymbol{\alpha})}{p(\mathbf{t}|\boldsymbol{\alpha}, \sigma^2)} \tag{8}$$

The solution is those relevance vectors where $\alpha_i < \delta$ and δ is a given threshold to filter those vectors where the α_i hyperparameter converges to infinity.

III. Social behavior and the oxytocine receptor

Beside the RVM research I am involved in a cooperative research project with the Semmelweis University (SE) and the Eötvös Loránd University (ELTE). We examine genes related to social behavior. First, we analyzed oxytocine receptor gene in three species: humans, wolves and dogs. We tried to describe the connection between OXTR and social behavior. Initially, I searched for available knowledge in the NCBI and Ensemble database where the human and the dog gene sequence can be found, but there is no public genome for wolves. Using this information I carried out an in silico study to compare the human and dog oxytocine receptor gene and the protein. The biggest difference between the dog and the human OXTR is a 5 amino acid long deletion in the dog OXTR. I also built a phylogenetic tree as you can see in Figure 1. The oxytocine receptor gene is a quite good indicator for clustering species. For detailed information read [3].

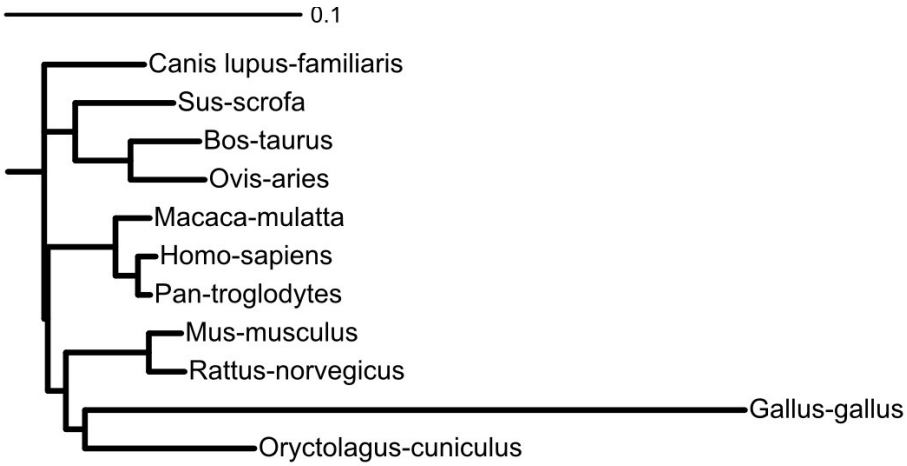


Figure 1: Phylogenetic tree [3]

The second task is to find other genes related to social behavior using Endavour [4]. I used Endavour for two training sets. In the first set, I used the genes which are connected to social behavior in Gene

Ontology [5]. This training set includes 27 genes. In the second case, I used a training set for all genes that were used by the research group at SE in former studies including 32 genes. The training set is based on experts' knowledge about social behavior. There is an overlap between the two training sets, so I am expecting similar results. As we do not have candidate genes, I ran the Endavour on the full human genome in both cases. As you can see on the left side in Table 1 after the fusion of results for each database but Gene Ontology in Endavour, we get the expected outcome. In the first case MAPK8IP1, NLGN1 and DLG2 while in the second case DRD5 are the only ones which were not included in the training set. Because of running on the full genome the first 20 highest ranked genes contains almost exclusively the training genes. We have to analyze the results for all genes with experts to validate the influence of these genes on social behavior. The analysis helps to find candidates. With these candidate genes together with some control genes, I can validate and complete the set of genes.

Table 1: Endavour results for the two different training sets. The results for the training set from Gene Ontology are on the left and the results from the second training set (SE gene set) are shown on the right hand site

symbol	Global prioriti-zation score	rank	rank ratio	symbol	Global prioriti-zation score	rank	rank ratio
HRAS	1	2.70e-16	0.000044	DRD2	1	1.39e-17	0.000044
KRAS	2	1.31e-14	0.0000879	DRD1	2	1.16e-16	0.0000879
DLG4	3	5.40e-13	0.000132	ADRA1A	3	5.37e-15	0.000132
MAPK8IP2	4	2.00e-12	0.000176	DRD3	4	5.89e-14	0.000176
DRD3	5	3.24e-12	0.00022	HTR2C	5	7.56e-14	0.00022
IL1B	6	1.22e-11	0.000264	ADRA2A	6	1.59e-13	0.000264
DRD4	7	4.08e-11	0.000308	HTR1B	7	1.65e-13	0.000308
NLGN3	8	1.28e-10	0.000352	BDNF	8	3.33e-13	0.000352
OXTR	9	1.98e-10	0.000396	NR3C1	9	4.65e-13	0.000396
MKKS	10	3.15e-10	0.00044	SLC6A3	10	5.14e-13	0.00044
TH	11	4.02e-10	0.000484	HTR2A	11	1.11e-12	0.000484
AVPR1A	12	6.62e-10	0.000528	CSNK1E	12	1.14e-12	0.000528
CHRNB2	13	1.14e-9	0.000572	DRD4	13	1.25e-12	0.000572
MAPK8IP1	14	5.50e-9	0.000616	DRD5	14	2.16e-12	0.000616
NLGN4X	15	8.45e-9	0.00066	EGLN2	15	3.88e-12	0.00066
NLGN1	16	9.98e-9	0.000704	OPRM1	16	1.27e-11	0.000704
DBH	17	1.49e-8	0.000748	SLC6A4	17	3.14e-11	0.000748
DLG2	18	3.42e-8	0.000792	COMT	18	4.35e-11	0.000792
AVP	19	8.97e-8	0.000835	MAOA	19	5.74e-11	0.000835
OXT	20	1.30e-7	0.000879	OXTR	20	1.19e-10	0.000879

Next I will run Endavour for the fused training sets and I will look for other training genes to get more general results. Another option is to perform a thematic search e.g. using a training set for the serotonergic genes. Another task is to use parts of these training sets for validation with Endavour. If I left out some genes, will the algorithm give these genes high rank? At the end I would like to have a set containing about 50 genes, which can be used in further studies. Last we would like to do a gene expression study to find SNPs and connect them to specific phenotype.

IV. Future Work

The next steps are the following. As I am traveling to the USA for a one-year research project in the next year I will concentrate on the oxytocine and the social behavior project. I will annotate the wolf OXTR gene and after finding new candidates for social behavior I will search for single nucleotide polymorphisms (SNP) in these genes. To carry out the research I will test SNP validation methods for next generation sequencers (NGS). In the RVM project the next step is to find a way to trace back the solution from the kernel space to the input space to make RVM for variable selection. The goal of this research is to extend the BayesEye so it can also handle continuous cases.

V. Conclusion

During last year I conducted two main researches. Both have some open questions. The relevance vector machine is a promising method which can be applied for Bayesian variable selection. Annotating and publishing the wolf genome is an interesting and challenging study. Getting deeper understanding of the social behavior of dogs and wolves can also lead to describe in more details fearful and aggressive behavior in humans. Last but not least, during this research project at the UCLA my task is to build long time cooperation between the Hungarian and the American research groups.

References

[1] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, 1(3):211–244, 2001, Cited By (since 1996): 1017.

[2] E. I. George and R. E. McCulloch, "Variable selection via gibbs sampling," *Journal of the American Statistical Association*, 88(423):881–889, Sept. 1993.

[3] P. Marx, A. Arany, Z. Ronai, P. Antal, and M. Sasvari-Szekely, "Genetic variability of the oxytocin receptor: an in silico sudy," *Neuropsychopharmacologia Hungarica*, 13(3):139–144, 2011.

[4] S. Aerts, D. Lambrechts, S. Maity, P. Van Loo, B. Coessens, F. De Smet, L.-C. Tranchevent, B. De Moor, P. Marynen, B. Hassan, P. Carmeliet, and Y. Moreau, "Gene prioritization through genomic data fusion," *Nat Biotech*, 24(5):537–544, May 2006.

[5] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S., Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, "Gene ontology: tool for the unification of biology," *Nature Genetics*, 25(1):25–29, May 2000.